Universidade de Aveiro
2016

Departamento de Eletrónica, Telecomunicações e Informática

Ricardo Jorge Lopes de Almeida

BICICLETA CONTROLADA - UMA QUESTÃO DE EQUILIBRÍO

Universidade de Aveiro
2016

Departamento de Eletrónica, Telecomunicações e Informática

Ricardo Jorge Lopes de Almeida

BICICLETA CONTROLADA - UMA QUESTÃO DE EQUILÍBRIO

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Eletrónica e Telecomunicações, realizada sob a orientação científica do Professor Doutor Rui Manuel Escadas Ramos Martins, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro

o júri

presidente

Prof. Doutor Ernesto Fernando Ventura Martins,
professor auxiliar, Universidade de Aveiro

Prof. Doutor Daniel Filipe Albuquerque
professor adjunto da Escola Superior de Tecnologia e Gestão de Viseu

Prof. Doutor Rui Manuel Escadas Ramos Martins
professor auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro

agradecimentos

É de salientar a importante colaboração da oficina de bicicletas, Garagem Paciência, sediada na cidade de Ovar, distrito de Aveiro, Portugal, e os mais sinceros agradecimentos a Humberto Lopes, tio do autor desta tese, pelos serviços prestados relativamente à construção, idealização e design de hardware do sistema físico da bicicleta.

palavras-chave

Bicicleta, equilíbrio, autónoma, controlo, direção, guiador, PID, PI, motor DC, servo-motor, Modelo Whipple, sensores, MPU6050, I2C, PIC32, PWM, microcontrolador, infravermelhos, velocidade, angulos, acelerómetro, simulação, parametros, hardware.

resumo

O presente trabalho propõe divulgar o estudo e a realização efetuada, de forma a obter um sistema composto por uma bicicleta, capaz de efetuar um passeio autónomo, em equilíbrio. Esta tese é constituída por uma apresentação dos componentes incluídos ao sistema inicial, onde figuram conceitos teóricos importantes à composição dos mesmos. É feita a descrição do modelo final relativamente ao sistema físico construído, onde este terá como objetivo a realização de uma viagem sem qualquer contacto físico com o utilizador, consistindo o sistema de equilíbrio na atuação exclusiva do guiador. A bicicleta é adaptada de forma a mover-se através da energia elétrica, implementando assim dois sistemas de controlo, equilíbrio e velocidade. O foco incide na criação de um sistema capaz de realizar o objetivo, para uma determinada velocidade. Os parâmetros reais da bicicleta foram medidos e utilizados para criar um ambiente de simulação da bicicleta em *Simulink*, onde vários controladores foram simulados antes da implementação no projeto de controlo da bicicleta física.

.

keywords

Bicycle, equilibrium, autonomous, control, PID, PI, servo-motor, DC motor, Whipple Model , sensors, MPU6050, I2C, PWM, PIC32, microcontroller, velocity, infrared, angles, acelarometer, simulation, parameters, hardware.

abstract

This current paper intends to reveal the study and the achievement made, in order to obtain a system composed of a bike, able to make, in balance, a separate walk. This thesis consist of a presentation of the components included in the initial system, which also comprise important theoretical concepts to their composition. The final model is described in relation to the physical system, where the final objective is to carry out a trip without any contact with an user, consisting in the exclusive operation of the handlebar to maintain the equilibrium. The bicycle is adapted to move through electric energy, thus implementing two systems of control, balance and speed. The focus is on creating a system capable of achieving the goal for a given speed. The actual bike parameters were measured and used to create a *Simulink* bike simulation environment where several controllers were simulated prior to implementation in the physical bike control project.

.

Índice

Lista de Figuras e Tabelas

Lista de Circuitos

Lista de Acrónimos

ADC	Conversor analógico-digital;
Bootloader	Programa de carregamento Software;
Brute Force	Tipo de algoritmo de descodificação pouco eficaz;
DMP	Processador de Movimento Digital;
Duty-cycle	Ciclo de trabalho, em percentagem;
GPS	Sistema de Posicionamento Global;
I2C	*Inter-Integrated Circuits*, Protocolo de comunicação;
IR	*Infra-red*, Infraestrutura de comunicação;
Lean_Phi	Inclinação atual da bicicleta;
LQR	Controlador *Linear quadratic regulator*;
MATLAB	Programa de cálculo e análise utilizado;
Master-Slave	Protocolo de comunicação;
MPLabX - IDE	Ambiente de programação;
OCM	Output Compare Module, geração *duty-cycle*;
Optocouplers	Isoladores óticos;
Over-engineered	Solução demasiado complexa para o problema;
Overshoot	Percentagem do pico máximo em relação ao valor final;
PD	Controlador Proporcional Derivativo
PI	Controlador Proporcional Integral
PID	Controlador Proporcional Integral Derivativo;
Polyval	Função *MATLAB* capaz de gerar funções através de pontos;
Ponte H	Controlador para motores DC, com travagem regenerativa;
Power-Bank	Bateria portátil;
Process Noise	Ruído adicionado involuntariamente ao processo;
Pull-Up	Tipo de agregação de resistências;
PuTTy	Programa de comunicação *Utilizador - Hardware*;
PWM	*Pulse With Modulation* - comunicação com motores;
RAW	Valores gerados sem unidades;
RPM	Rotações por minuto;
SCLx	Linha de comunicação do relógio, *I2C*;
SDAx	Linha de comunicação de dados, *I2C*;
Settling Time	Tempo que a resposta demora a chegar ao valor esperado;
Setpoint	Ponto desejado;
Signal Builder	Bloco gerador de sinais, Simulink;
Simulink	Programa incluído no *MATLAB* para simulações;
Sprocket	Roda dentada da bicicleta;
Steer_deltha	Ângulo final da direção da bicicleta;
Timer	Contador de tempo do microcontrolador;
T_Phi	Inclinação inicial/constante da bicicleta/estrada;
T_deltha	Ângulo da direção;

Capítulo 1

Introdução

1.1 Motivação

Um grande número de pessoas começa por aprender a andar de bicicleta com a ajuda de rodinhas, aquelas pequenas rodas que mantêm a bicicleta em equilíbrio, enquanto os utilizadores, sem conhecimento do perigo, ganham confiança e aumentam a velocidade entre uma ou outra queda. Estas pequenas rodas podem entrar para o esquecimento se for encontrado um sistema eletrónico de equilíbrio para estes veículos, uma nova forma de aprender a andar de bicicleta sem dispositivos adicionais, sendo agora a própria bicicleta responsável por equilibrar-se. Não seria fantástico e inovador ver uma bicicleta simplesmente a andar sozinha? Equipada com um sistema de equilíbrio, usando sensores e pequenos motores, pode-se concretizar a ideia, impedindo que a bicicleta perca o controlo. Contudo, numa fase inicial o sistema só funcionará adequadamente quando esta circula num percurso pré-determinado. O sistema inicial que possibilite as especificações anteriores consiste então na alteração do veio do guiador da bicicleta, modificando-a para incluir uma capacidade de gerar forças contrárias ao movimento de queda, através da rotação do servo adaptado no quadro deste veículo. Só esta correção não seria suficiente para o controlo de equilíbrio do sistema completo, sendo também necessário atuar num motor DC que permita o eficaz controlo da velocidade desejada. O circuito de controlo, incluindo microprocessador, componentes mecânicos e outros componentes adicionais, tais como uma bateria externa, e a adaptação necessária para transformar a bicicleta atual, num veículo movido a motor, poderão ser modificados e incluídos no projeto.

1.2 O Problema

O objetivo deste projeto de dissertação passa pela implementação de um sistema de equilíbrio através para uma bicicleta simples de criança. Não só o equilíbrio é importante para esta problemática, pois é necessário ter em conta a transformação desta bicicleta normal, para uma capaz de se movimentar através de um motor elétrico. Para tal, será necessário executar uma abordagem teórica ao problema, onde através da criação de um modelo matemático se efetuarão simulações, antes de dar o passo para o mundo real. É possível retratar uma visão geral das etapas para atingir este objetivo. Dividindo o projeto em 3 partes distintas, dependentes entre si, de seguida apresenta-se uma visão de alto nível do trabalho descrito neste documento.

1.2.1 Primeira Parte - Simulação

A primeira parte deste projeto considera a criação de um modelo matemático que possa ser utilizado num ambiente de simulação. Uma ferramenta que possuí esta capacidade é o *MATLAB* [1], onde é possível efetuar diversas simulações ao sistema com o objetivo de encontrar um modelo capaz de controlar o equilíbrio de uma bicicleta em movimento. O *MATLAB* [1] torna-se um programa útil nestas situações por possuir a ferramenta *Simulink* que possibilita a construção de um modelo de blocos capaz de simular o sistema físico. Para recriar o que foi mencionado, é necessário compreender a estrutura física da bicicleta, nomeadamente, a forma como esta se comporta em movimento, ou seja, aquilo que pode ser designado por momento de inércia. A partir destas noções é fundamental encontrar um modelo que as conjugue e que seja passível de implementação num ambiente de simulação, como é o caso do *Simulink*.

1.2.2 Segunda Parte - Montagem

Relativamente à segunda parte, a montagem do setup experimental pode ser subdividida em dois pontos distintos, sendo o primeiro correspondente à montagem física de um servo no quadro da bicicleta capaz de movimentar o guiador para ambos os lados. O segundo ponto é equivalente à transformação da tração normal da bicicleta, para uma bicicleta movida através de um motor elétrico. Ambas as alterações anteriores vão ser geridas por uma unidade inteligente com microcontrolador, eletrónica de interface e fonte de alimentação.

1.2.3 Terceira Parte - Programação e Teste

Relativamente à última fase, esta passa pela criação do código computacional e software adequado ao suporte físico disponível, consoante as normas e ideias implementadas, quer na primeira como na segunda parte deste projeto. Após esta conclusão entra-se na fase de testes ao sistema com o objetivo de ajustar, tanto os componentes físicos implementados, como o código e parâmetros programados no microcontrolador. Finalmente, o projeto entrará em fase de conclusão, onde a bicicleta deverá efetuar o respetivo passeio autónomo, em velocidade e equilíbrio.

1.3 State of the Art

Nesta secção, de seu nome "*State of the Art*", irá ser apresentado, tal como o próprio nome indica, o estado de desenvolvimento geral de sistemas semelhantes ao conceito deste projeto de dissertação. Sendo referido de igual modo o nível de desenvolvimento alcançado por esses mesmos sistemas, explicitando assim as técnicas e metodologias apresentadas para a realização dos mesmos. É possível assumir esta secção, como um ponto de partida e de exclusão de ideias, considerando os diversos modelos realizados, e assim tentar formular uma solução inovadora para o problema apresentado, tomando uma perspetiva criteriosa sobre os problemas encontrados nas diversas ideias já existentes. Ao longo dos últimos anos, muitos trabalhos têm sido efetuados usando para isso uma bicicleta. Estes projetos, relacionados com controlo de equilíbrio nestes sistemas, foram iniciados na década de 80, podendo concentrar-se no trabalho realizado por Loftum [2] e Bjermeland [3]. Este último focou-se no trabalho com a dinâmica de bicicletas e modelagem matemática, enquanto Loftum [2] desenvolveu a instrumentação e sistemas de computador para estas. Desde 2006, o sistema de computador e instrumentação tem sido desenvolvido e corrigido por Fossum [4], Sølvberg [5], Brekke [6] e Hatlvoll [7].

1.3.1 Autonomous Bicycle

Autonomous Bicycle trata-se de um projeto de mestrado de Dag Christian Ånnestadum, aluno pertencente à *Norwegian University of Science and Technology* [8], em que desenvolveu um sistema de equilíbrio para uma bicicleta, baseada num modelo com um pêndulo invertido. O ponto fulcral desse projeto passou então por uma bicicleta que, após a aplicação de um controlo adequado, é capaz de executar um passeio autónomo, usando o já mencionado, pêndulo invertido. Após ter sido criado um controlador que torna a bicicleta capaz de realizar esse percurso, o foco passou por não ser usado o controlador mais complexo, mas sim um sistema simples que cumpra o objetivo. Foi criado um ambiente de simulação da bicicleta em *Simulink,* testando variados controladores para implementar na bicicleta física. Como pode ser verificado pela descrição efetuada, os objetivos de *Autonomous Bicycle* [8], assemelham-se em larga escala aos pretendidos nesta dissertação, apresentando no entanto uma solução diferente, recorrendo a outros tipos de composição de hardware, nomeadamente na atuação mecânica do guiador, funcionando este com o recurso a um complexo sistema de engrenagens.

1.3.1a Motivação para a realização do Autonomous Bicycle

O trabalho no desenvolvimento de uma bicicleta autónoma com recurso a um sistema de controlo proporcionado por um pêndulo invertido começou a ser desenvolvido por volta da década de 1980, tendo-se verificado nos últimos anos, algum desenvolvimento neste sistema de bicicleta, mas sem grandes sucessos relativamente à construção de uma bicicleta autónoma. Tornou-se então objetivo do projeto *Autonomous Bicycle* [8] obter uma bicicleta capaz de realizar um percurso em auto equilíbrio. O conceito de controlo de uma bicicleta com um pêndulo invertido que simule um ciclista que se inclina é também um assunto interessante que foi adaptado à questão do referido projeto.

1.3.1b Problemas e adaptação ao projeto

Este trabalho efetuado, torna-se um excelente ponto de comparação para a realização do projeto de dissertação que aqui irá ser explicitado, uma vez que os objetivos são bastante semelhantes, criando assim alguns conceitos interessantes para iniciar e desenvolver soluções mais adequadas ao projeto em si.

1.3.2 Jyrobike

Este projeto trata de um conceito bastante interessante, tendo um número elevado de referências, por ser um produto já comercializado pela internet. Refere-se então a uma bicicleta com um sistema de autorregulação de equilíbrio e como é indicado pelos próprios, perfeita para pessoas com certos tipos de deficiência, ou crianças com uma maior dificuldade em aprender a andar de bicicleta. Surgiu assim a ideia do *Jyrobike* [9], cujo funcionamento é baseado nas ciências e propriedades do giroscópio.

1.3.2a Conceito do Sistema de Controlo

O sistema de controlo deste modelo está concentrado na roda dianteira da bicicleta sendo este capaz de gerar uma força de estabilização que resiste à força de tombar devido à sua inclinação ou da simples gravidade. Esta força é gerada através do momento angular gerado por massas circulares que se encontram a girar a uma certa velocidade. É então possível diminuir as configurações de equilíbrio à medida que o ciclista se vai sentindo mais confiante, podendo este sistema ser controlado remotamente por tecnologias wireless. O sistema atinge assim vários pontos de equilíbrio, dependendo do próprio peso do piloto e da capacidade de condução da bicicleta, podendo estes variar consoante as diferentes velocidades de rotação das massas internas, gerando momentos angulares diferentes. O sistema pode ser exemplificado pelas figuras *1.1 e 1.2*, onde são analisados muito simplificadamente os diversos componentes deste sistema implementado na roda da frente da bicicleta. Uma vez que este se encontra centralizado

apenas num componente, é possível uma rápida instalação e adaptação a qualquer bicicleta, dependendo apenas do tamanho da roda.

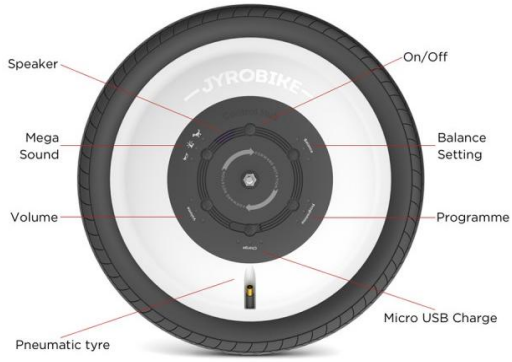


Figura 1.1 - Roda dianteira da bicicleta Jyrobike [9]

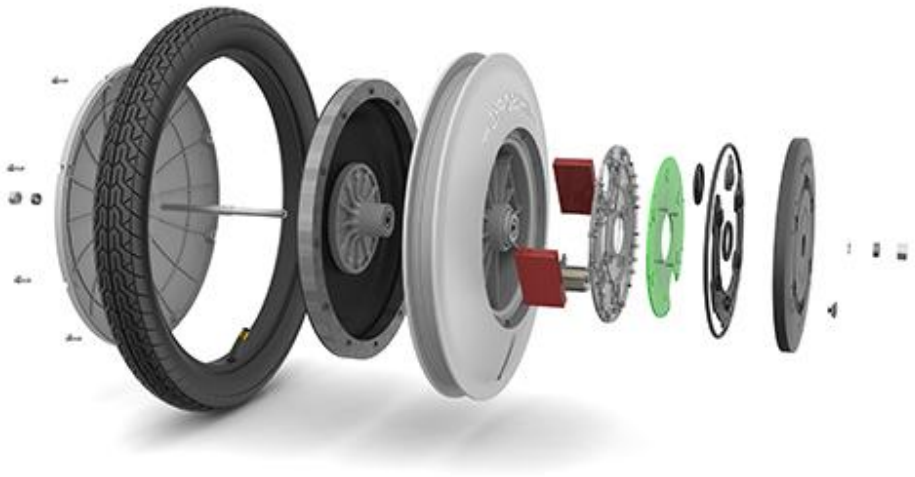


Figura 1.2 - Sistema de controlo de equilíbrio Jyrobike [9]**;**

1.3.2b Problemas e adaptação ao projeto

Este sistema encontrado pode proporcionar uma abordagem interessante à tentativa de solucionar o problema de controlo de equilíbrio de uma bicicleta. Apesar de funcionar com várias aplicações mais complexas, o sistema central de equilíbrio baseia-se num modelo e conceito simples de introduzir num projeto, contudo não se insere no cerne desta tese, uma vez que o conceito é adaptar uma bicicleta simples, com a finalidade de executar um percurso sem auxílio de forças externas ao sistema, como um piloto. Sendo que este sistema foi desenhado para ajudar um possível utilizador, serão úteis recolher alguns conceitos de controlo de equilíbrio de forma a adaptar uma bicicleta simples com o objetivo desta andar, num percurso controlado, sozinha.

1.4 Visão Geral

Neste ponto pretende-se fazer uma listagem dos capítulos presentes nesta dissertação, dando assim uma ideia dos temas que irão ser abordados.

- O Capítulo 2 dará uma breve descrição das possíveis estruturas de hardware e software a implementar no início desta tese, bem como uma descrição da teoria de fundo relevante para o desenvolvimento desta tese.
- O Capítulo 3 descreve o trabalho realizado no âmbito de hardware e software.
- O Capítulo 4 descreve os métodos e trabalhos realizados para medir os parâmetros necessários para criar um modelo da bicicleta física.
- O Capítulo 5 descreve o desenvolvimento de um modelo com os parâmetros medidos no Capítulo 4 em conjunto com o desenvolvimento de modelos e simulação de controladores para o equilíbrio.
- O Capítulo 6 descreve a implementação e resultado do controlador de equilíbrio e de velocidade no sistema real.
- O Capítulo 7 descreve soluções futuras que possam ser incluídas ao sistema já criado.
- O Capítulo 8 apresenta uma discussão sobre o resultado do modelo físico e simulações, assim como, comentários às escolhas feitas durante esta tese e conclusão desta.

Capítulo 2

Teoria e Possíveis Implementações

Este capítulo irá fazer uma primeira abordagem ao tema, procurando soluções e módulos a aplicar ao sistema, por forma a torná-lo completo e independente da ação humana. Serão então descritas algumas ideias iniciais para a sua implementação, mesmo que parte delas não se tenham materializado na solução final. Servem então de guia para a resolução da problemática criada e a concretização de um sistema eficiente e robusto que cumpra o objetivo. Com o objetivo de desenvolver um sistema de controlo adequado a todas as necessidades do cerne deste projeto, é fundamental possuir e descrever um modelo matemático do sistema em questão. Como o modelo básico se trata realmente de uma bicicleta, para o efeito, existem já estudos bastante aprofundados sobre as diferentes variantes do mesmo. O modelo pelo qual se irá optar é o mais usado em diversos estudos relativos a simulações de bicicletas, de seu nome *Whipple Bycicle Model* [10]. Baseia-se num conjunto de equações lineares derivadas por *Papadopoulos* [11], sendo o conjunto de equações disponíveis mais estudadas neste âmbito. Irá então ser descrito o modelo desenvolvido por *Whipple* [10], explicando com isso as equações lineares obtidas por *Papadopoulos* [11] e os seus parâmetros necessários, de forma a completar o modelo.

2.1 Sistema de Comunicação Remota

Um possível módulo a implementar no sistema geral é o de um componente de rádio transmissão de dados via rádio. Trata-se de uma solução simples que através de um microcontrolador adaptado, utilize uma plataforma de desenvolvimento com um sistema de comunicação por radiofrequência, ou *Bluetooth*. Este módulo poderá ser utilizado com o intuito de controlar a velocidade da bicicleta. Este sistema pode ser reutilizado numa outra fase, implementando usos diferentes para a transmissão, tais como, alterar o caminho pré-definido e receber informações relevantes dos sensores da bicicleta.

2.2 Sistema de Tracking e Trem de Aterragem

Outro módulo passível de ser implementado no sistema base da bicicleta é o de algo capaz de mapear o deslocamento real da bicicleta, habilitando assim o sistema a definir um percurso predeterminado para a bicicleta percorrer. Isto pode ser realizável assumindo no entanto que existe um módulo semelhante a um trem de aterragem, que

efetua o controlo de equilíbrio por hardware específico até se atingir a velocidade pré definida. Mais uma vez, refira-se que são elementos extras, podendo estes módulos ser considerados após a finalização do objetivo principal.

2.3 Constituição de Hardware

A solução pensada inicialmente, passa pelo rascunho de um modelo físico apresentado na *Figura 2.1*. Este sistema de hardware para a realização do problema consiste num modelo onde a velocidade de uma bicicleta elétrica é atribuída *à priori* por software, e controlada através do seu respetivo controlador externo para motores DC. Em relação ao equilíbrio desta, esse está a cargo da atuação no veio da forqueta. O guiador é então movido a partir de um servo-motor, estando este conectado por algum tipo de conetor rígido mas ajustável. Assim o sistema geral será possivelmente capaz, através de um microcontrolador programado corretamente, de proporcionar um deslocamento autónomo em equilíbrio da bicicleta.

Figura 2.1 - Sistema simplificado Hardware [12]**;**

2.3.1 Microcontrolador

O microcontrolador a ser usado no projeto é o de um *PIC32* [13]. Este sendo da empresa *microship* [13] é fornecida gratuitamente uma plataforma de desenvolvimento capaz de gerar o código necessário à programação do mesmo. O microcontrolador possui, para além de outras importantes características, módulos de comunicação I2C e SPI, que permitem a ligação a acelerómetros, sensores de temperatura, entre outros. O microcontrolador comunica com o sistema computacional através de um módulo USB. É

uma solução interessante e talvez a mais adequada, já que o autor desta tese se encontra familiarizado com a utilização deste microcontrolador.

2.3.2 Servo-Motor

Para o servo foi pensado a utilização de um modelo semelhante aos utilizados em aero-modelismo, com uma relação preço/qualidade razoável. Como características essenciais consideram-se o facto de ser necessário possuir engrenagens de metal, importantes para a robustez do sistema e um binário suficiente para virar o volante da bicicleta. Existe no mercado uma oferta bastante elevada, portanto será presumivelmente fácil encontrar um que respeite as necessidades requeridas ao projeto.

2.4 Soluções de Software

O software a ser utilizado no desenvolvimento deste projeto, passará por ambientes de programação adequados ao microcontrolador escolhido, bem como algum ambiente que forneça a capacidade de realizar simulações a um modelo matemático. Para este ultimo ponto, como já referido, o *MATLAB* [1] e a ferramenta nele incluída, *Simulink*, irão ser os escolhidos.

2.4.1 Desenvolvimento

O desenvolvimento de código fonte para o microcontrolador pode ser desenvolvido em MPLAB IDE [14] que se trata de um programa de software executado num PC com a finalidade de desenvolver aplicações para microcontroladores microchip e controladores de sinal digital. Trata-se assim do chamado *Integrated Development Environment* (IDE), porque fornece um único ambiente integrado para desenvolver o código para microcontroladores embutidos.

2.5 Teoria de Controlo

Para o objetivo principal de equilíbrio e velocidade é necessário a introdução de algoritmos de controlo capazes de gerar a compensação adequada ao sistema. Irá ser então descrita neste ponto alguma teoria importante à questão, que irá ajudar na compreensão e resolução do problema apresentado. Alguma desta teoria pode ser considerada como conhecimento básico, mas é fundamental explicitar algumas noções importantes para facilitar a compreensão das soluções apresentadas a pessoas não familiarizadas com a teoria utilizada. Em [8] são apresentadas mais soluções e

explicações mais desenvolvidas em relação ao resumo aqui apresentado da mesma referência.

2.5.1 Representação em Espaço de Estados

A representação em espaço de estados é uma maneira de descrever um sistema físico através de um conjunto de entradas, saídas e variáveis de estado como um modelo matemático. Os sistemas lineares e invariantes no tempo podem ser descritos na forma:

$$\dot{x} = Ax + Bu \quad (2.1a)$$
$$y = Cx + Du \quad (2.1b)$$

Onde 'x' é o vetor das variáveis de estado 'n' dimensional, 'A' um matriz constante 'n' por 'n' , 'u' vetor das entradas 'r' dimensional e 'B' matriz constante 'n' por 'r', e 'y' o vetor das saídas.

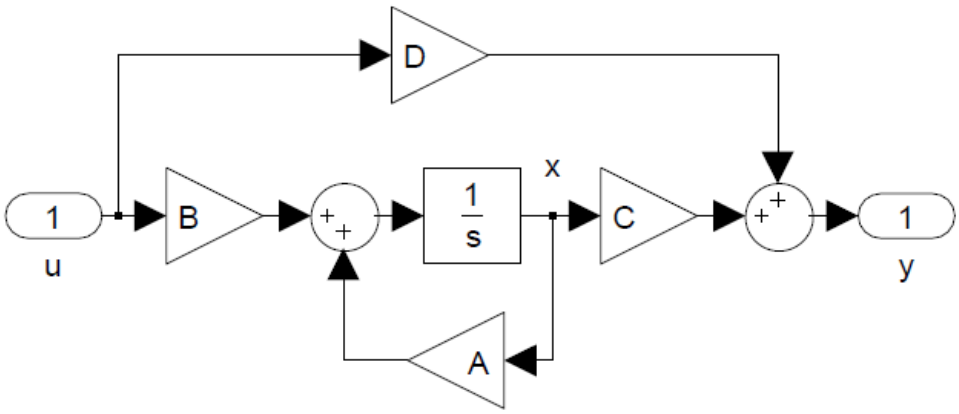


Figura 2.2 - Representação em Espaço de Estados [8]**;**

2.5.2 Do controlador P até ao PID

O controlador PID [15] é dos controladores disponíveis mais estudados e utilizados, facilmente descrito na *Figura 2.3*. Deve notar-se que durante a sua implementação, alguns tópicos precisam de ser levados em conta. Uma primeira anotação a ser observada trata-se do fenómeno *integral windup*, que ocorre quando existe uma grande mudança no set-point, gerando um termo integral maior do que o sinal máximo de controlo para o processo, provocando oscilações no sistema. Com isto, deve ser implementado alguma forma de *anti-windup*, limitando o valor máximo do integral. Um segundo ponto a ter em conta passa pelo termo da derivada, onde pequenas quantidades de ruído nas medidas podem causar grandes mudanças na saída. Deve ser então utilizado algum método para a remoção de ruído durante o projeto do controlador, como por exemplo, efetuar uma filtragem passa-baixo conveniente, onde se remova o ruído mas não altere a dinâmica do sistema.

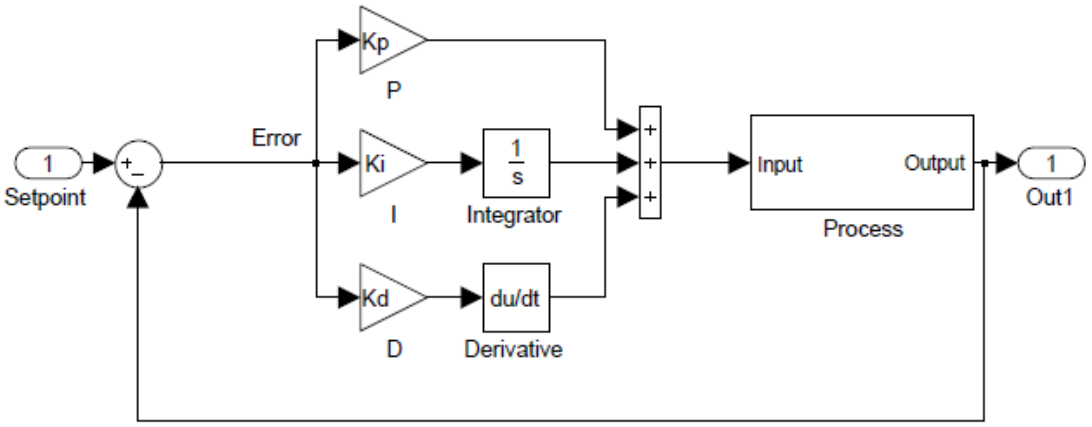


Figura 2.3 - Controlador PID [8]**;**

2.6 Modelo Matemático da Bicicleta

Neste subcapítulo o sistema em questão é analisado de forma bastante teórica, abordando assim um modelo matemático complexo de uma estrutura simples, com vista a poder recriar em ambiente de simulação de computador, a física inercial desta mesma estrutura. Em [8] e [10] são apresentadas em detalhe as obtenções das equações descritas bem como a teoria que as acompanha. Nesta secção apenas serão apresentadas as equações finais e algumas restrições e condicionantes mais relevantes ao sistema.

2.6.1 Modelo da Bicicleta segundo Whipple

É possível assumir o modelo de bicicleta segundo *Whipple* [10] como o modelo mecânico de uma bicicleta simples, considerando a ausência de um piloto, ou simplesmente, a existência estática deste. Consiste basicamente num sistema dividido por quatro partes diferentes, a saber, quadro da bicicleta, incluindo a massa e estática do piloto; o conjunto, guiador e forqueta; a roda traseira e a roda da frente da bicicleta. Estes quatro subsistemas são interligados entre si, formando o modelo da bicicleta.

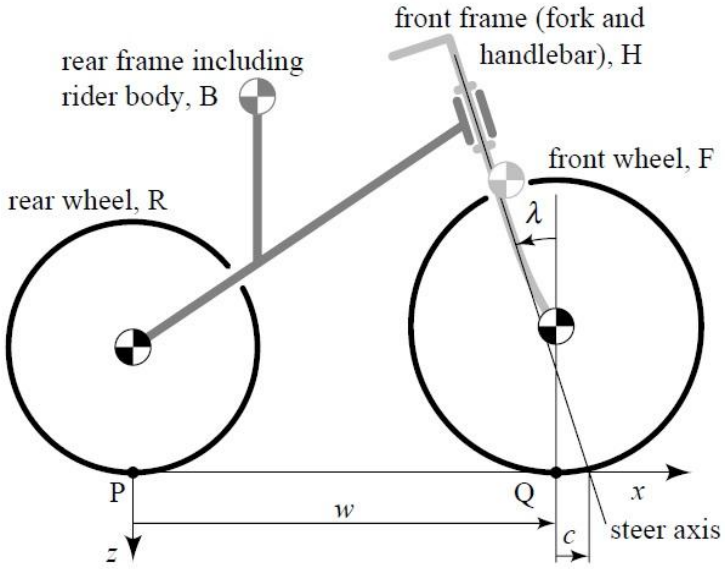


Figura 2.4 - Whipple Bicycle model [10]**;**

O modelo escolhido assume algumas restrições específicas que serão aqui descritas por virtude do rigor que se pretende obter nas simulações que se irão efetuar.

- *O modelo assume que ambas as rodas estão em contacto com o solo e que o este é completamente horizontal.*
- *O modelo assume que existe simetria em relação ao plano XZ quando a bicicleta está de pé com a direção na posição neutra e simetria circular das rodas.*
- *O modelo assume rodas do tipo knife-edge sem bordos arredondados.*
- *O modelo assume que, para ser considerado um piloto este terá de ser um piloto estático, ou seja, o modelo não assume qualquer movimento de um piloto no que diz respeito ao quadro da bicicleta.*
- *O modelo assume uma estrutura 100% rígida.*
- *O modelo não assume qualquer forma de amortecimento.*
- *O modelo não assume a existência de forças de atrito nas ligações entre peças.*
- *O modelo assume um tipo de pneu incapaz de escorregar ou derrapar.*

O modelo é reduzido quando se remove as quatro restrições *não-holonómicas* [1] relacionadas com os pontos de contacto roda-terra (longitude e lateral para cada contacto). A bicicleta pode então ser descrita por apenas três parâmetros, sendo eles:

- *A taxa de inclinação do corpo traseiro (quadro) Φ.*
- *A taxa de direção δ.*
- *A taxa de rotação da roda traseira ΦR em relação à estrutura do quadro.*

[1] Definem-se como não-holonómicos sistemas com dimensão finita onde algum tipo de restrição é imposta a um ou mais estados do sistema. Estas limitações podem ser provocadas pela conservação do momento angular, condições impostas pela impossibilidade de deslocar numa ou mais direções, como resultado da imposição de restrições durante o projeto do sistema de controlo, pelo fato do sistema não possuir atuadores em todas as direções do espaço do problema, entre outros.

Considera-se que:

- *Virar à direita e inclinação para a direita são considerados como as direções positivas.*
- *Movimentar a bicicleta para a frente é considerado a direção positiva para ΦR.*

É possível obter-se as equações[2] linearizadas de movimento para a direção e inclinação (*δ*, *Φ*), que não são mais do que um conjunto de duas equações diferenciais ordinárias com coeficientes constantes de segunda ordem, acopladas. Estas duas equações são combinadas, *equação 2.2*, com uma força externa aplicada, *f* = (T*Φ* e T*δ*) do lado direito da equação.

$$M\ddot{q} + vC_1\dot{q} + [gK_0 + v^2K_2]q = f \tag{2.2}$$

Os coeficientes linearizados das equações são:

- ***q*** = $[\Phi,\ \delta\ ]^T$ *correspondente às variáveis variantes no tempo, angulo de inclinação e angulo de direção.*
- ***f*** = $[T\Phi,\ T\delta\ ]^T$ *correspondente às variáveis variantes no tempo das forças de inclinação e de direção.*
- ***M*** *é a matriz de massa e momento de inércia, que introduz no modelo a energia cinética.*
- ***C1*** *é a matriz de amortecimento, Damping Matrix, que captura os sinais do giroscópio devido à ação de virar a direção da roda e do angulo de inclinação. Ela também captura as reações inerciais devido à taxa da direção.*
- ***K0*** *é a matriz de rigidez independente da velocidade, que introduz a energia potencial no modelo.*
- ***K2*** *é a matriz de rigidez dependente da velocidade, que introduz no modelo o efeito giroscópico e as forças centrífugas.*

De forma a descrever os diferentes coeficientes das equações linearizadas, são necessários os 25 parâmetros descritos na *tabela 2.1*. A respetiva medição e estimativa destes será descrita posteriormente.

[2] Estas equações, bem como as que irão ser apresentadas no seguimento deste capitulo, foram obtidas com a ajuda de um trabalho semelhante, que serviu assim de guia para a adaptação a este projeto, *Autonomous Bicycle* [8].

Parâmetros	*Símbolo*
Base da Roda	**ω**
Deslocamento	**c**
Desl. Eixo de Direção	**λ**
Gravidade	**g**
Velocidade em Frente	**v**
Roda Traseira	**R**
Raio	r_R
Massa	m_R
Momento Inércia	$(I_{Rxx}, I_{Ryy}, I_{Rzz})$
Quadro	**B**
Centro de Massa	(x_B, z_B)
Massa	m_B
Momento Inércia	$\begin{bmatrix} I_{Bxx} & 0 & I_{Bxz} \\ 0 & I_{Byy} & 0 \\ I_{Bxz} & 0 & I_{Bzz} \end{bmatrix}$
Conjunto da Frente	**H**
Centro de Massa	(x_H, z_H)
Massa	m_H
Momento Inércia	$\begin{bmatrix} I_{Hxx} & 0 & I_{Hxz} \\ 0 & I_{Hyy} & 0 \\ I_{Hxz} & 0 & I_{Hzz} \end{bmatrix}$
Roda da Frente	**F**
Raio	r_F
Massa	m_F
Momento Inércia	$(I_{Fxx}, I_{Fyy}, I_{Fzz})$

Tabela 2.1 - Descrição dos 25 parâmetros;

Com estas definições e variáveis irão formar-se as matrizes pretendidas para obtenção do modelo da bicicleta. A primeira trata-se, da matriz de massa, contendo os momentos de massa de inércia.

$$M_{\phi\phi} = I_{Txx}; \quad M_{\phi\delta} = I_{A\lambda x} + \mu I_{Txz} \tag{2.3a}$$

$$M_{\delta\phi} = M_{\phi\delta}; \quad M_{\delta\delta} = I_{A\lambda\lambda} + 2\mu I_{A\lambda z} + \mu^2 I_{Tzz} \tag{2.3b}$$

$$M = \begin{bmatrix} M_{\phi\phi} & M_{\phi\delta} \\ M_{\delta\phi} & M_{\delta\delta} \end{bmatrix} \tag{2.4}$$

Onde I_T e I_A representam o momento de massa e produto de inércia relativo ao ponto de contato da roda traseira e conjunto da frente, e μ o *mechanical trail ratio*.

A matriz rigidez dependente da gravidade:

$$K0_{\phi\phi} = m_T z_T; \quad K0_{\phi\delta} = -S_A \tag{2.5a}$$

$$K0_{\delta\phi} = K0_{\phi\delta}; \quad K0_{\delta\delta} = -S_A \sin\lambda \tag{2.5b}$$

$$K0 = \begin{bmatrix} K0_{\phi\phi} & K0_{\phi\delta} \\ K0_{\delta\phi} & K0_{\delta\delta} \end{bmatrix} \tag{2.6}$$

Onde m_T e z_T representam a massa total do sistema e a coordenada z do centro de massa da bicicleta, S_A o conjunto das forças devido à mudança de posição do centro de massa do quadro e guiador.

A matriz rigidez dependente da velocidade:

$$K2_{\phi\phi} = 0; \quad K2_{\phi\delta} = \frac{S_T - m_T z_T}{w}\cos\lambda \tag{2.7a}$$

$$K2_{\delta\phi} = 0; \quad K2_{\delta\delta} = \frac{S_A + S_F \sin\lambda}{w}\cos\lambda \tag{2.7b}$$

$$K2 = \begin{bmatrix} K2_{\phi\phi} & K2_{\phi\delta} \\ K2_{\delta\phi} & K2_{\delta\delta} \end{bmatrix} \tag{2.8}$$

Onde S_T e S_F representam os coeficientes giroscópios para as duas rodas.

E por fim a matriz de amortecimento (*Damping matrix*):

$$C1_{\phi\phi} = 0; \quad C1_{\phi\delta} = \mu S_T + S_F \cos\lambda + \frac{I_{Txz}}{w}\cos\lambda - \mu m_T z_T \tag{2.9a}$$

$$C1_{\delta\phi} = -(\mu S_T + S_F \cos\lambda); \quad C1_{\delta\delta} = \frac{I_{A\lambda z}}{w}\cos\lambda + \mu\left(S_A + \frac{I_{Tzz}}{w}\cos\lambda\right) \tag{2.9b}$$

$$C1 = \begin{bmatrix} C1_{\phi\phi} & C1_{\phi\delta} \\ C1_{\delta\phi} & C1_{\delta\delta} \end{bmatrix} \tag{2.10}$$

Capítulo 3

Descrição da Bicicleta, Hardware e Software

Neste capítulo pretende-se efetuar a descrição final do sistema físico, composto pela bicicleta e elementos adicionais implementados para a resolução do problema. Para tal, esta análise irá visar a composição de hardware e software do sistema. Este estudo irá proporcionar os detalhes do processo para a medição de parâmetros, com o intuito de obter o modelo de bicicleta *Whipple* [10], para o sistema utilizado, tal como descrito no *Capítulo 2*. De referir que os circuitos apresentados foram desenhados recorrendo à ferramenta *Eagle* [16], que proporcionou de igual modo o layout para impressão dos mesmos.

3.1 Composição de Hardware

O sistema final construído como protótipo para a resolução do cerne desta tese, é constituído pela adaptação de uma bicicleta regular para criança. Esta, composta por uma estrutura central, a que designamos quadro, ou corpo da bicicleta, as rodas dianteira e traseira, e uma estrutura frontal, denominada assim por ser referida ao conjunto forqueta e guiador. O sistema final pode ser visto na *Figura 3.1.* A adaptação da bicicleta simples requereu a inclusão de determinados componentes, em algumas partes desta. Irá ser então efetuada uma descrição sumária do hardware adicional que equipou o sistema, uma vez que este irá fazer parte integral do modelo final e das medições a serem realizadas.

Figura 3.1 - Protótipo Final do Sistema Bicicleta em Equilíbrio;

A primeira etapa da decomposição de hardware, baseada na interpretação do objetivo final, permite a sistematização do projeto através da realização do diagrama de blocos de todo o sistema. Na *Figura 3.2* apresenta-se o diagrama de blocos geral, a partir do qual se implementou, na prática, o sistema. Para esta implementação física, é necessário introduzir componentes essenciais ao funcionamento do mesmo. Assim, como pode ser visto, obtém-se dois sistemas de controlo autónomos, onde um é responsável por controlar a velocidade da bicicleta, imposta *à priori,* e outro, com o objetivo de ajustar a posição do volante consoante a inclinação da bicicleta. De modo a garantir estes parâmetros de controlo são então ajustados para a velocidade escolhida e estimados em modelos de simulação, que podem ser analisados no *Capitulo 5*.

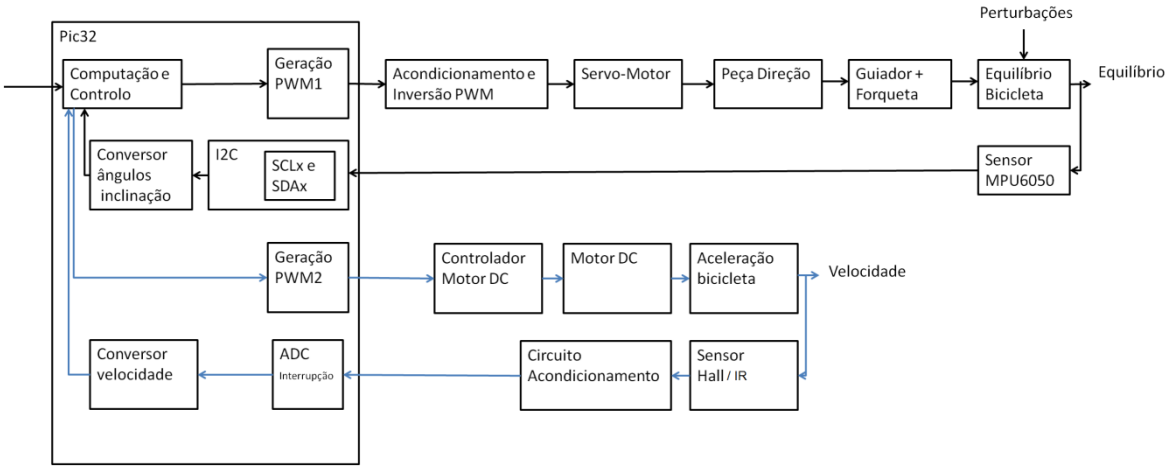


Figura 3.2 Diagrama de Blocos do Sistema Completo;

3.1.1 Servo-Motor

O equilíbrio da bicicleta estará dependente da atuação do volante da mesma, *Figura 4.5*. Para tal, é fundamental a inclusão de um componente capaz de gerar binário suficiente e controlado, de modo a corrigir a direção, atuando no veio do conjunto forqueta e guiador. Um servo-motor *hexTronic HX12K* [17] foi utilizado no projeto. Este, constituído por rolamentos metálicos, possuí a robustez necessária, podendo gerar um binário de 12kg/cm quando alimentado com 6V. Para além destas características, o servo apresenta velocidade de operação bastante satisfatória, de 0.20s/60graus a 4.8V. Controlado através de pulsos *PWM*, torna-se bastante acessível a sua adaptação ao sistema.

3.1.1.1 Circuito Inversão PWM

Para que o servo-motor consiga funcionar de um modo adequado, este necessita de receber pulsos *PWM*, com tensões de pico na ordem dos 3 a 5V. Dado que o microcontrolador utilizado apresenta apenas tensões de trabalho na ordem dos 0-3.3V, é

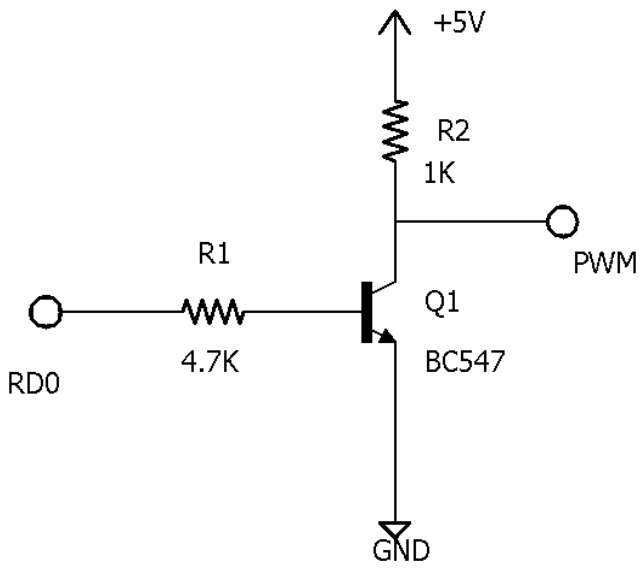
necessário a criação de um bloco de acondicionamento. Este é usado para a mudança de nível do sinal *PWM*, sendo constituído pelo transístor Q1 e pelas duas resistências R1 e R2, *Circuito 3.1*. O circuito funciona como um interruptor em que o transístor comuta entre a zona de corte e de saturação. As resistências assumem os valores calculados para uma corrente de base de 0.4mA e um ganho em corrente de 10A/A por forma a garantir a saturação do transístor.

$$R_1 = \frac{V_{in\,min} - V_{be}}{I_B} \tag{3.1}$$

$$R_1 \frac{2.6-0.7}{0.4m} = 4750\Omega \approx 4.7K\Omega,\ com\ V_{in\,min} = V_{OH\ PIC32} = 2.6V \tag{3.2}$$

$$R_2 = \frac{V_{DC} - V_{CEsat}}{I_B * h_{fesat}} \tag{3.3}$$

$$R_2 = \frac{5-0.2}{0.4m*10} = 1200\Omega \approx 1K\Omega \tag{3.4}$$



Circuito 3.1 - Circuito Acondicionamento e Inversão PWM - Servo;

O modo de funcionamento *PWM* do servo-motor baseia-se em pulsos enviados pelo microcontrolador a cada 20ms. Um pulso com duração a '1' de 1ms ou 2ms faz o servo-motor rodar para as suas extremidades, numa gama de -90 a 90 graus. Um pulso a '1' de 1.5ms reposiciona o servo na sua posição neutra. O nível de tensão deste sinal, enviado pelo microcontrolador, tem o valor [0..0,4]V para '0' e [2,6..3,3]V para nível '1'. Já que o

circuito implementado, amplifica e inverte o sinal, é necessário a emissão deste sinal invertido, ou seja, enviar pulsos com duração a '0' de 1ms ou 2ms de forma a movimentar o servo para as suas extremidades. Assim o pulso PWM após inversão e mudança de nível apresenta o valor [0..0,2]V para '0' e [4,8..5]V para nível '1', sendo as tensões de pico necessárias para o correto funcionamento deste. A *Figura 3.3* representa a forma como o tON foi obtido.

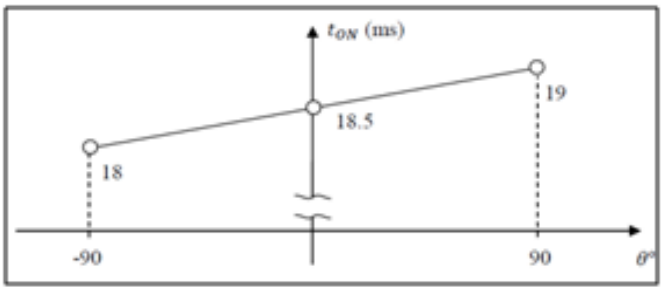


Figura 3.3 - Tempo a 1 para o servo-motor;

No entanto, devido às limitações físicas do sistema, a gama de atuação do servo motor é mais reduzida, produzindo apenas uma gama de aproximadamente [-70º,70º].

3.1.1.2 Peça de Direção

A ligação entre o servo-motor e o veio do conjunto guiador e forqueta é efetuada através de uma peça desenhada especificamente para o projeto. Esta baseia-se num design de direção de karts e foi adaptada por forma a movimentar a direção da bicicleta através do binário do servo. Os dois elementos encontram-se conectados entre si por dois eixos rotativos e um elemento rígido ajustável, para o correto acoplamento do sistema. A *Figura 3.4* representa a peça metálica da direção, montada num ângulo de 25º para que adicione uma maior excursão de movimentos ao servomotor. O elemento rígido que efetua esta ligação não passa de um simples tirante, ou barra de aço com cerca de 0.18m, com duas roscas em ambas as pontas, permitindo assim a inserção de parafusos que proporcionem aperto, mas com uma ligeira folga. A ligação ao tirante está situada a uma distância de 0.12m permitindo assim atenuar o binário necessário para mover a direção.

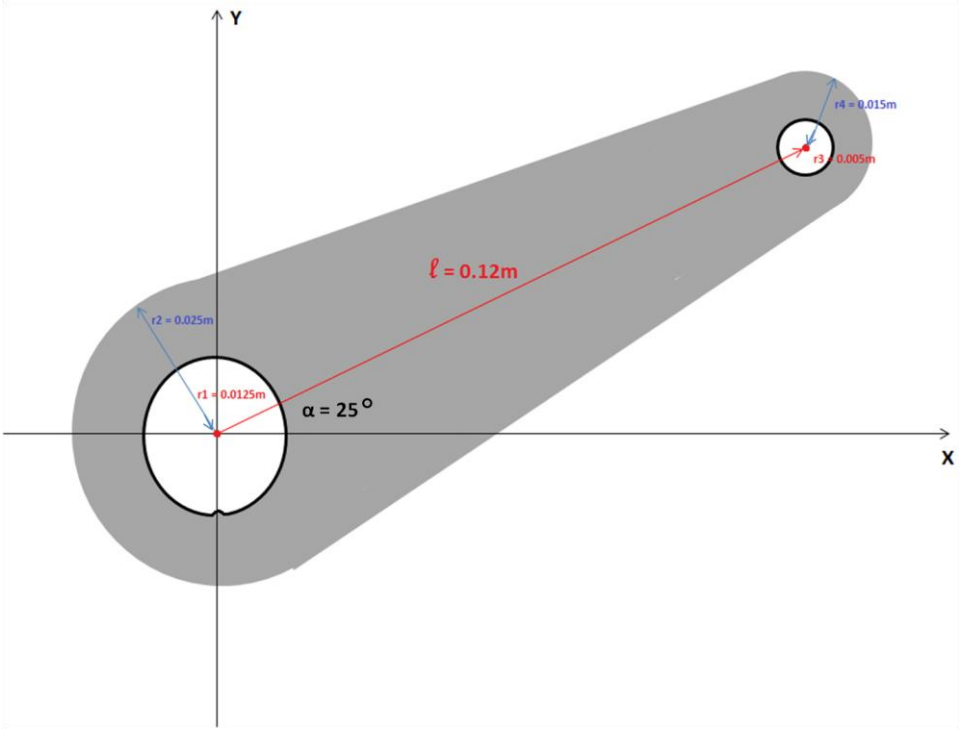


Figura 3.4 - Peça da direção;

Devido ao angulo criado por este conjunto de direção, os ângulos que são criados pelo servo não correspondem aos obtidos no veio da direção. Assim, estes foram medidos e interpolados em *MATLAB* (comando *polyval*), obtendo uma equação aproximadamente linear, *equação 3.5,* que relaciona o angulo criado pelo servo, *X,* e o angulo provocado na direção, *Y*.

$$Y = 0.576\,x + 1.104 \tag{3.5}$$

Figura 3.5 - Sistema de direção;

3.1.2 Motor Elétrico

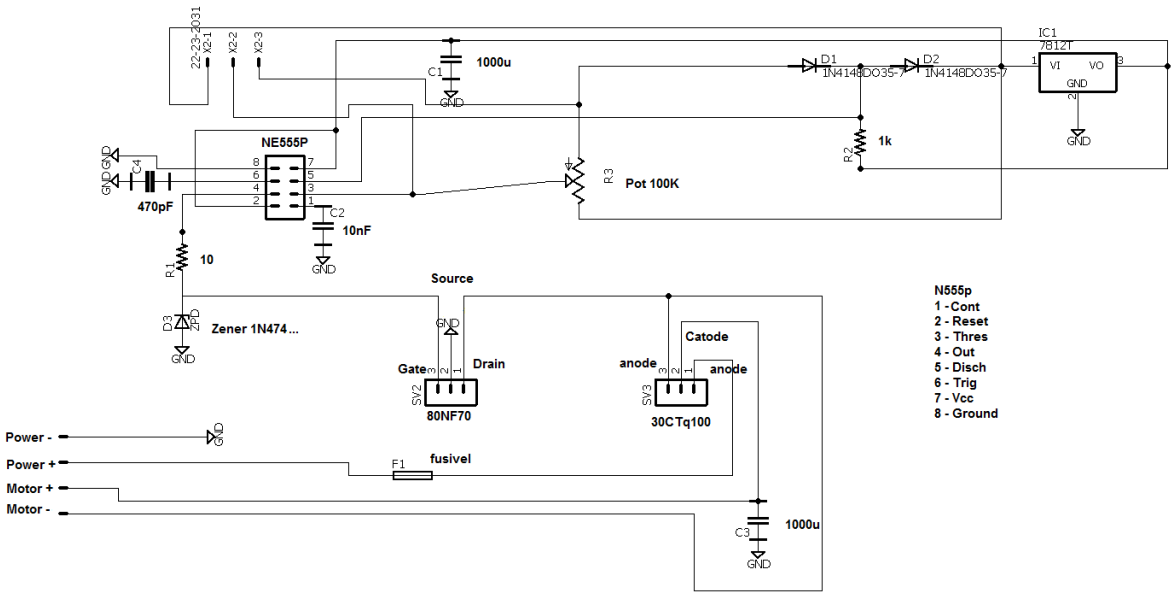
Este é o elemento capaz de acelerar a bicicleta, tornando-a independente da tração humana. Instalado no veio metálico da parte inferior do quadro, o motor elétrico, adequado à instalação em *scooters,* possui uma potência máxima de 250 Watt, podendo ser alimentado por uma fonte de 24V e 14A. Com estas características o motor tem capacidade para atingir, em modo livre, 2750 rotações por minuto. Contudo a sua aplicação na bicicleta necessitou de certas alterações, devido ao pequeno tamanho e espaço reduzido da mesma, encontrando-se aquele apenas a funcionar a metade da sua potência máxima. Para a fixação e integração no sistema foi necessário introduzir certos componentes adicionais.

3.1.2.1 Roda Dentada com passo 6.35mm

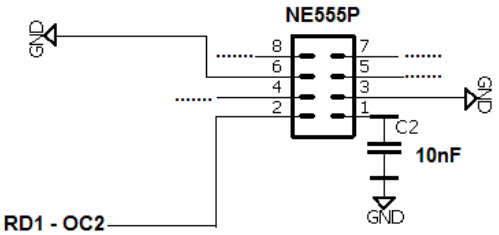
Para que o motor possa proporcionar a tração desejada à bicicleta, é necessário a utilização de uma roda dentada com um passo mais preciso, com o valor de 6.35 mm. Esta, também denominada por *sprocket,* é conectada ao veio de tração da roda dentada da bicicleta inicial. A ligação entre o motor e esta nova *sprocket* é efetuada através de uma corrente com o mesmo passo entre os elos, nomeadamente 6.35 mm. A roda traseira e a roda dentada inicial, presente na pedaleira, encontram-se ligadas através de uma corrente de passo normal.

3.1.2.2 Controlador para Motor DC

O correto funcionamento do motor necessita da aplicação de um sistema adicional que o possa controlar. Neste caso, foi utilizado um controlador para motores DC, que utiliza sinais *PWM*, *pulse with modulation,* para realizar o controlo da velocidade do motor utilizado. Assim será possível controlar a velocidade da bicicleta gerando este tipo de sinais através de um potenciómetro incluído na placa. O esquema geral deste *driver* do motor está representado no *Circuito 3.2a*. Contudo, será necessária a alteração do circuito, uma vez que se pretende que o sinal PWM seja gerado pelo microcontrolador e não pelo potenciómetro, para que seja possível efetuar o controlo do *duty cycle* a aplicar consoante a velocidade instantânea. O *Circuito 3.2b* pretende representar essa adaptação do timer NE555P [18].



Circuito 3.2a - Circuito geral do Controlador Motor DC;



Circuito 3.2b - Adaptação Timer NE555P;

Sumariamente o circuito inicial, *Circuito 3.2a,* tem como base o funcionamento de um timer configurado para enviar pulsos à frequência de 13Khz. Estes, são modulados no tempo através da atuação de um potenciómetro gerando assim sinais PWM a serem enviados ao motor. O estudo do *datasheet* do timer NE555P [18] levou à alteração no circuito por forma ao pulso enviado para o motor, seja formado no microcontrolador. Assim o timer NE555P [18] adaptado, *Circuito 3.2b,* passa a funcionar como um buffer que recebe o sinal PWM do microcontrolador, porto RD1, e o envia para o motor.

3.1.3 Microcontrolador

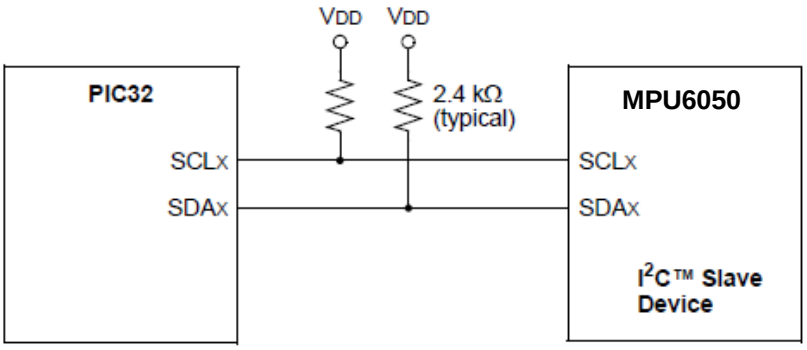
Para que se automatize o sistema, é fundamental a inclusão de um microcontrolador. O escolhido foi o *PIC32MX775F512H* [13], dada a familiarização do autor com este microcontrolador e pelo fato de este possuir uma capacidade de cálculo elevada quando comparada com microcontroladores de 8 e 16 bits.

3.1.4 MPU6050

A medição da inclinação da bicicleta terá de ser efetuada através de um sistema capaz de obter ângulos a partir de uma posição inicial. O *MPU-6050* [19] é um dispositivo de *MotionTracking* de 6 eixos, que combina um giroscópio de 3 eixos, um acelerómetro de 3 eixos e um processador de movimento digital, *DMP.* Este possui três conversores analógico-digitais de 16 bits, *ADCs*, para digitalizar as saídas do giroscópio e três *ADCs* de 16 bits para digitalizar as saídas do acelerómetro. Para uma análise em precisão dos movimentos rápidos e lentos, as partes apresentam uma escala programável pelo utilizador de ± 250, ± 500, ± 1000 e ± 2000°/s, para o giroscópio, e uma gama de ± 2 g, ± 4g, ± 8g, e ± 16g para o acelerómetro. A comunicação com todos os registos do dispositivo é realizada através de comunicação I2C a 400kHz. O *MPU-6050* opera numa gama de tensão de 2.375V-3.46V. Para além disto, o dispositivo proporciona um pino de referência VLOGIC, que define os níveis lógicos da sua interface I2C. A tensão pode assumir VLOGIC 1.8V ± 5% ou VDD. De notar que o PIC32 [13] pode implementar uma comunicação I2C, sendo assim, diretamente compatível com o *MPU-6050* [19].

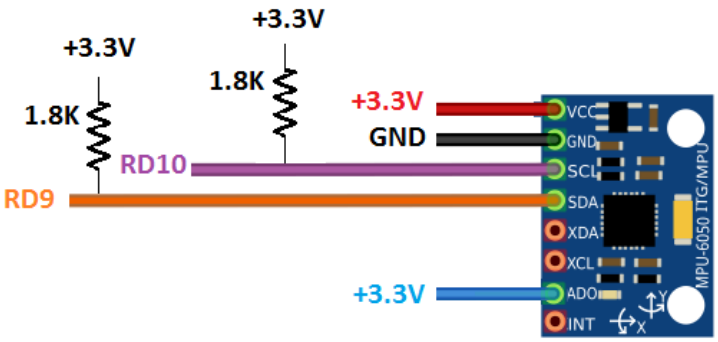
3.1.4.1 Circuito Comunicação I2C

As comunicações I2C [13] requerem um circuito de adaptação para que possam ser efetuadas corretamente. A interface emprega um protocolo abrangente para garantir a transmissão e receção de dados. Nesta comunicação, um dispositivo torna-se o "master", iniciando a transferência no barramento e gera os sinais de relógio para possibilitar a transferência. O outro dispositivo atua como *slave*, respondendo à transferência. A linha do relógio, *SCLx*, é a saída do "master" e entrada para o *slave.* A linha de dados, *SDAx*, pode ser a saída e a entrada, tanto do "master" como do *slave*. Devem ser então inseridos resistências de *pull-up* externas, sendo estas usadas para garantir um nível elevado quando nenhum dispositivo está a colocar a linha para baixo. Assim a comunicação terá de ser efetuada de forma semelhante ao *Circuito 3.3.*



Circuito 3.3 - Comunicação I2C;

O *Circuito 3.4* pretende representar a ligação estabelecida entre o MPU6050 e os portos de entrada/saída do microcontrolador, por comunicação I2C.



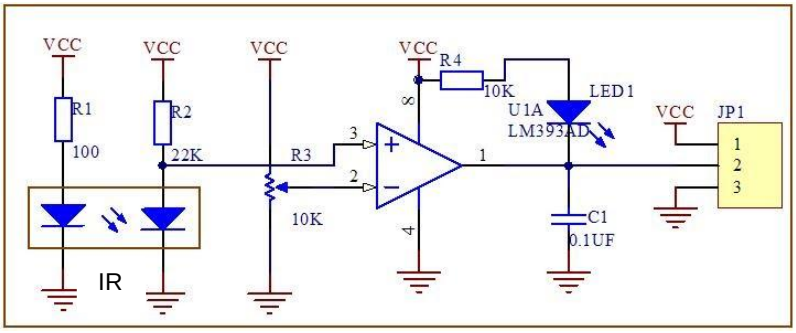
Circuito 3.4 - Comunicação PIC32 - MPU6050;

3.1.5 Sensores de Velocidade - Tacómetros

Com a necessidade da medição em tempo real da velocidade instantânea, foram pensados e implementados dois tipos de sistemas no modelo geral da bicicleta. Estas soluções tratam-se de um sensor de Hall adaptado e de um sensor infravermelhos. No entanto, na constituição do modelo final, apenas este último foi colocado em funcionamento e por isso apenas este será aqui descrito.

3.1.5.1 Sensor de Velocidade Linear

Foi adquirido um módulo composto por uma fenda de infravermelhos e um comparador LM393 [20]. Este sistema é de fácil compreensão, onde é emitido um pulso para o microcontrolador assim que algo intercetar o feixe IR. Assim, este módulo foi adaptado ao quadro da bicicleta, contando as interceções dos dentes da roda dentada da pedaleira, determinando as rotações por minuto e por consequência, a velocidade linear. *O Circuito 3.5* representa um esquema genérico do sensor utilizado, onde estão representados os díodos fotelétricos que enviam os sinais para o comparador aquando da interceção do feixe IR. A *Figura 3.5* representa este módulo introduzido no sistema final.



Circuito 3.5 - Circuito genérico Sensor de Velocidade Linear [21]**;**

Figura 3.5 - Sensor Velocidade IR;

3.1.6 Sistema de Alimentação

Para que todos os componentes possam funcionar, é fundamental instalar um sistema de alimentação, capaz de produzir todas as diferentes tensões necessárias. Duas baterias de 12V e 5A/H são acopladas ao sistema de modo a obter duas alimentações isoladas entre si. A primeira fonte irá alimentar o controlador do motor DC, que por sua vez proporciona a tensão necessária ao motor de propulsão. A segunda bateria está ligada a uma cascata de reguladores de tensão para alimentação dos componentes do circuito geral.

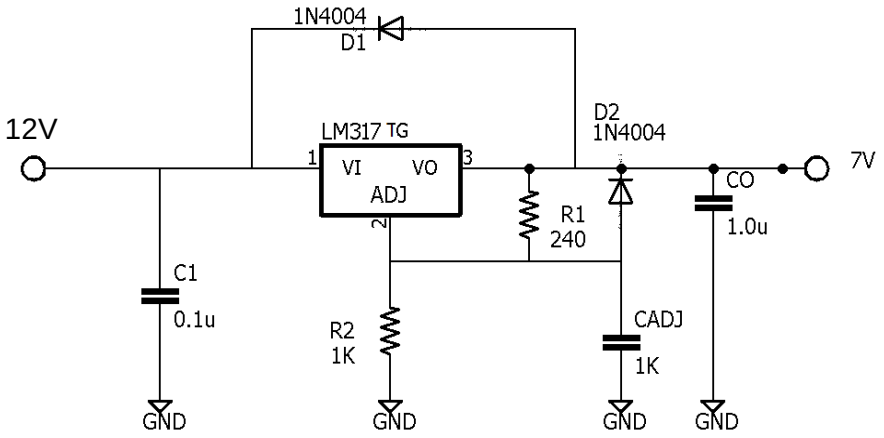
3.1.6.1 Circuitos Reguladores de Tensão

De modo a beneficiar do máximo binário possível do servo-motor, este será alimentado com a sua tensão máxima, nomeadamente 7V. Para tal efeito, foi utilizado um regulador de tensão ajustável LM317TG [22], que a partir de uma tensão de entrada de 12V pode gerar uma larga gama de tensões de saída, através de um canal de ajuste. O circuito utilizado para gerar esta tensão de 7V, está representado no *Circuito 3.6.*

$$V_o = V_{REF}\left(1 + \frac{R_2}{R_1}\right) + \left(I_{ADJ} * R_2\right) \tag{3.6}$$

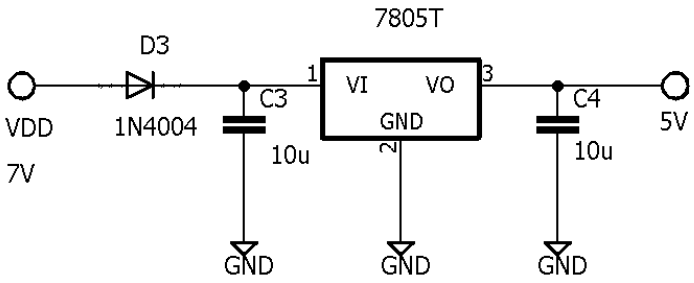
$$7 = 1.25 * \left(1 + \frac{R_2}{240}\right) + (50u * R_2) \tag{3.7}$$

$$R_2 \approx 1K\Omega \tag{3.8}$$



Circuito 3.6 - Regulador de Tensão 7V;

Os circuitos de acondicionamento e inversão do sinal de PWM necessitam de uma tensão de alimentação de aproximadamente 5V. Esta tensão é conseguida através de um regulador de tensão LM7805C, ligado em cascata, que a partir dos 7V gerados pelo LM317TG, gera os 5V necessários a estes circuitos.



Circuito 3.7 - Regulador de Tensão 5V;

3.1.6.2 Power-Bank

Foi considerado importante, que o microcontrolador tivesse igualmente uma alimentação independente dos restantes componentes, por forma a garantir a ausência de ruído e interferências externas que possam ser transferidas pelos canais de alimentação. Para resolver esse problema poderiam ser utilizados *optocouplers,* mas por opção técnica, evitando assim a integração de mais circuitos, recorreu-se à utilização de um simples *power-bank,* utilizado no quotidiano para o carregamento de dispositivos *USB.*

3.1.7 Suporte e Outros

Por forma a acomodar os componentes utilizados na construção do modelo, desde as baterias e sistemas de reguladores, ao microcontrolador e circuitos adjacentes, foi construído um suporte sobre a roda traseira, capaz de transportar segura e eficazmente todos os componentes descritos. Como pode ser visto na *Figura 3.6*. Foram também introduzidos alguns pequenos módulos de ventilação, botões de ligação e suporte adicional de prevenção para quedas, composta pelas rodas de apoio levantadas a uma certa distância do chão, não permitindo assim uma inclinação crítica da bicicleta, durante a realização de testes de funcionamento, *Figura 3.7.*

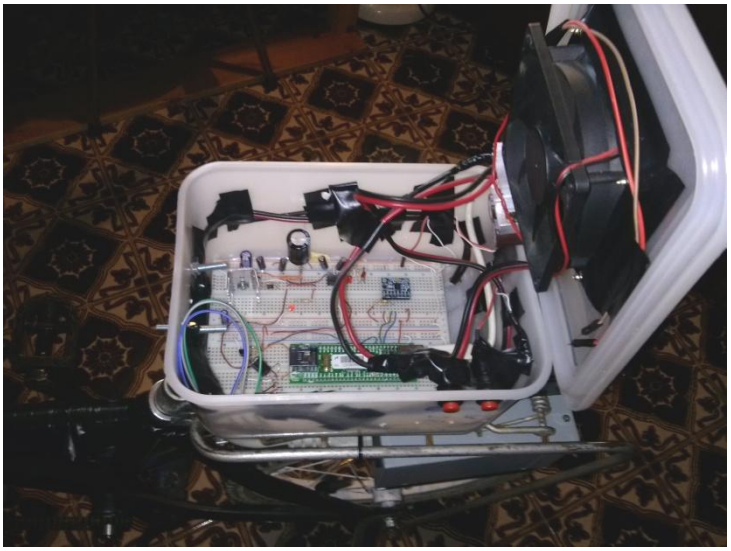


Figura 3.6 - Suporte e módulos instalados;

Figura 3.7 - Rodas de apoio;

3.1.8 Layout PCB's

Com o avançar do projeto procedeu-se ao desenho do circuito elétrico completo utilizando para isso a ferramenta *Eagle* [16]. Com esta, foi possível obter o layout para a implementação do circuito utilizando PCB's, *Figura 3.8,* no entanto a sua utilização foi descartada por o modelo físico final se tratar de um protótipo, utilizando então uma placa branca para a sua construção.

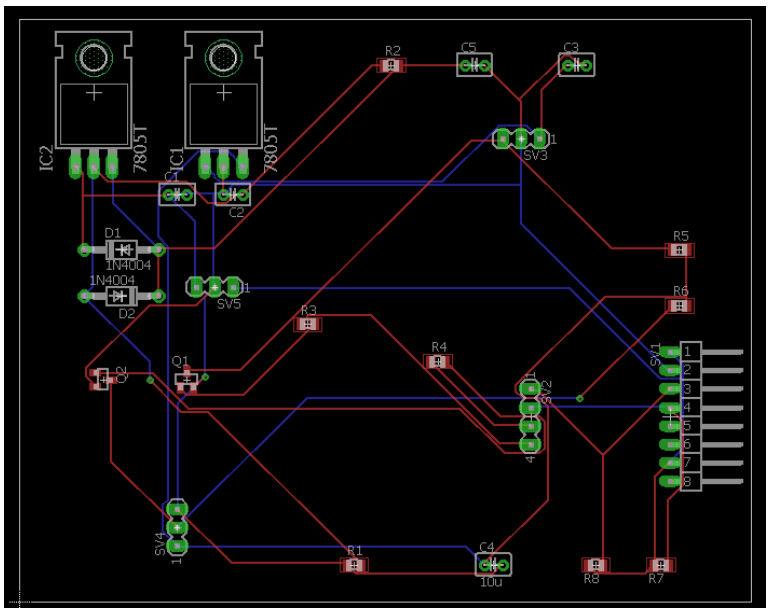


Figura 3.8 - Layout PCB do Circuito Elétrico;

3.2 Constituição de Software

O sistema final de hardware foi descrito sumariamente no *ponto 3.1,* e com isto, é possível ter a perceção dos recursos de software necessários ao correto funcionamento deste projeto. Para além de programas capazes de efetuar simulações de modelos, como o *MATLAB e Simulink*, é imprescindível a utilização de software de programação para microcontroladores, que para além da escrita de código, possibilite a instalação e comunicação com o mesmo, de uma forma simples e prática. Como não é funcional, efetuar o teste de equilíbrio numa fase inicial, diretamente ao sistema, foram criadas pequenas funções de teste, que permitem verificar as comunicações entre os blocos. Estas irão ser descritas resumidamente, pois apenas se tratam de funções acessórias ao projeto. O software do sistema final irá ser descrito e explicado no *capítulo 5* desta tese.

3.2.1 MPLabX

O MPLabX é um IDE, *Integrated Development Environment*, ou seja, um software de edição que recria um ambiente de programação com o intuito de desenvolver aplicações para microcontroladores, tal como a PIC32 utilizada neste projeto. É fornecido gratuitamente pela empresa *Microchip Technology* e assim foi integrado neste projeto como o editor de código em linguagem C.

3.2.1.1 Bootloader

O Bootloader trata-se simplesmente da aplicação que carrega o ficheiro *hex,* criado pelo compilador do MPLabX, no microcontrolador. Esta estabelece a comunicação com a memória da Pic32, podendo formatá-la e programá-la consoante desejado, através da ligação USB estabelecida.

3.2.1.1 PuTTy

O programa PuTTy, utilizado na fase de testes, resulta na aplicação que executa uma comunicação de entrada/saída com o utilizador e microcontrolador, utilizando para isso, recursos do próprio, como a UART, device driver da PIC32 responsável por este tipo de comunicação.

3.2.2 Funções de Teste

Por forma a efetuar o teste de comunicação com os diferentes blocos do sistema final, foram criadas pequenas funções de validação, tanto para a leitura dos valores recebidos pelo sensor *MPU6050* [19], como para emissão dos sinais PWM para o servo-motor. De igual modo foi desenhada uma função capaz de agregar estes dois tipos de comunicação. O código completo destas funções pode ser analisado e facilmente compreendido no Apêndice A1.

Capítulo 4

Medição dos Parâmetros da Bicicleta, Modelo *Whipple*

Neste capítulo pretende-se efetuar uma descrição pormenorizada da medição dos diferentes parâmetros do sistema, composto pela bicicleta e elementos adicionais implementados para a resolução do problema. Para tal, irá ser tido em conta o modelo de *Whipple* [10] já explicitado, proporcionando assim a obtenção dos parâmetros utilizados na criação de um modelo de simulação. Para estas medições foram criadas algumas ferramentas úteis que irão, de igual modo, ser descritas. A *Figura 4.1* pretende representar o modelo da bicicleta utilizada, constituída pelos respetivos parâmetros.

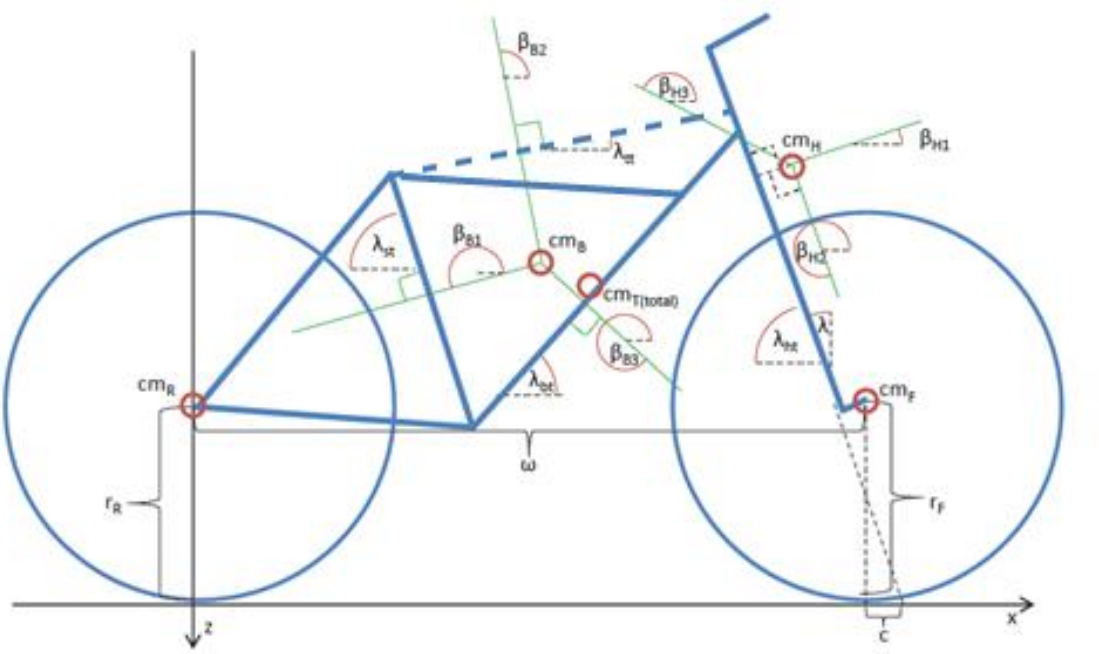


Figura 4.1 - Diferentes Parâmetros da Bicicleta [23]**;**

4.1 Raio da Roda

O valor do raio da roda foi obtido medindo a distância que cada roda percorre durante 15 rotações. As medições de comprimento foram realizadas utilizando uma fita de métrica com uma resolução de 1 milímetro e uma precisão de medição estimada de ± 10mm. As

medidas foram realizadas com os pneus a uma pressão de aproximadamente 35 psi. O raio da roda foi calculado com a seguinte fórmula em que *d* é a distância medida:

$$r = \frac{d}{2\pi n} \tag{4.1}$$

O comprimento percorrido pela roda da frente foi de 18.235m e a roda de trás obteve uma distância percorrida de 18.274m. Os raios da roda da frente e da roda traseira foram calculados usando a *equação 4.1.*

$$r_F = 0.1935\ m \tag{4.2}$$

$$r_R = 0.1939\ m \tag{4.3}$$

4.2 Ângulos do Quadro

Os ângulos do quadro foram medidos manualmente utilizando um transferidor apoiado num nível de bolha, de forma a garantir a exatidão destas medições. A bicicleta foi apoiada verticalmente, de forma perpendicular ao chão, usando para isso um suporte construído previamente com esse intuito. De referir também que os pneus da mesma se encontravam com uma pressão de aproximadamente 35 psi. O transferidor utilizado tinha uma resolução de um grau, assim sendo estas medições devem ser consideradas tendo uma precisão de ±1 grau. Os ângulos do quadro foram medidos para:

$$\lambda_{ht} = 59^o \tag{4.4}$$

$$\lambda_{tt} = 13^o \tag{4.5}$$

$$\lambda_{st} = 72^o \tag{4.6}$$

$$\lambda_{bt} = 46^o \tag{4.7}$$

A inclinação do eixo de direção é o complemento do ângulo da forqueta, λ_{ht} ,e pode ser calculada a partir dessa medição:

$$\lambda = 31^o \tag{4.8}$$

Após esta medição a bicicleta foi apoiada num suporte que tenta representar a inclinação imposta pelas rodas, *Figura 4.2*, com os pneus desta a uma pressão de aproximadamente 35 psi.

Figura 4.2 - Quadro Simples da bicicleta apoiado num suporte;

4.3 Deslocamento da Forqueta

Para calcular o deslocamento da forqueta foram efetuadas medições recorrendo a outro equipamento de medida que possibilitasse valores com uma resolução superior. Para tal, um paquímetro com uma resolução de 0,05 milímetros foi utilizado. A medição foi realizada com a forqueta sobre uma mesa nivelada, de maneira a que o eixo de direção fosse paralelo à mesa. Assim foram medidas a distância a partir da mesa até ao centro da forqueta e a distância desde da mesa até ao centro do eixo de medição, *fo*. Relativamente à precisão das medições, pode ser considerada como sendo de 0,05 milímetro. Sabendo que o valor medido para a distância da mesa até ao centro da forqueta é de 30 milímetros obteve-se assim *fo* = 14 mm. O deslocamento foi calculado a partir a seguinte fórmula.

$$c = \frac{r_F \sin\lambda - f_o}{\cos\lambda} \tag{4.9}$$

$$c = 0.115\ m \tag{4.10}$$

4.4 Distância entre Eixos

A distância entre eixos corresponde à distância entre o centro das rodas e foi medido diretamente utilizando uma fita métrica com uma resolução de 1 milímetro e uma precisão de ± 1 milímetro.

$$w = 0.755\ m \tag{4.11}$$

4.5 Massa dos conjuntos

As massas do conjunto forqueta e guiador, m_H, rodas dianteira e traseira, m_F e m_R respetivamente, e quadro, m_B, foram medidas recorrendo a uma balança digital com uma resolução de 1 g e precisão de ± 1 g. A partir deste ponto irão de igual modo ser considerados os módulos e componentes adicionados à bicicleta, que compõem o sistema final, afetando assim a massa destes, bem como o centro de massa e respetivo momento de inércia.

$$m_H = 1.606\ kg \tag{4.12}$$

$$m_F = 1.134\ kg \tag{4.13}$$

$$m_R = 1.340\ kg \tag{4.14}$$

$$m_B = 7.915\ kg \tag{4.15}$$

4.6 Centro de Massa

Os centros de massa para as quatro partes da bicicleta irão ser determinados neste subcapítulo. Os cálculos nesta secção foram obtidos com a ajuda da ferramenta *MATLAB*, cuja rotina pode ser encontrada no *Apêndice B.1*, sendo esta uma adaptação para este protótipo, da rotina descrita no trabalho referenciado por [8].

4.6.1 Rodas

Para estar em conformidade com o modelo escolhido para as simulações, modelo de bicicleta *Whipple* [10], a localização do centro de massa das rodas foi considerada no centro geométrico destas. No sistema de coordenadas globais, os centros de massa das rodas irão ser descritos pelas seguintes equações:

$$cm_F = [x_F \quad y_F \quad z_F] = [w \quad 0 \quad -r_F] \tag{4.16a}$$

$$cm_R = [x_R \quad y_R \quad z_R] = [0 \quad 0 \quad -r_R] \tag{4.16b}$$

$$cm_F = [0.755 \quad 0 \quad -0.1935] \tag{4.17a}$$

$$cm_R = [0 \quad 0 \quad -0.1939] \tag{4.17b}$$

4.6.2 Quadro

Relativamente ao centro de massa do quadro da bicicleta, este correspondente ao corpo do sistema, foi obtido a partir de medições realizadas ao longo da experiência onde se utilizou um pêndulo de torção. O quadro foi então pendurado em três posições distintas, que por sua vez, foram baseadas em três ângulos: ao ângulo obtido na parte superior do tubo, o ângulo do tubo do selim e por fim o angulo do tubo paralelo inclinado até ao solo. Estes ângulos podem ser vistos e analisados através da *Figura 4.1*. Estes foram medidos tendo por base um nível, de forma a assegurar que o ângulo de deslocamento horizontal estaria perto de zero. A precisão dos ângulos, medidos com uma ferramenta adaptada que incluía um transferidor e um nível de bolha, é considerada estar dentro do intervalo ± 1º horizontalmente. Foi medida a distância horizontal, a_B,entre o eixo traseiro e a extensão do pêndulo de torção. O centro de massa é então calculado olhando para o eixo do pêndulo como uma linha no sistema de coordenadas *XZ*, com uma inclinação, *m*, uma interseção no eixo Z, *b* e assumindo um ângulo entre o eixo global X e o braço do pêndulo, onde se considera o sentido positivo para baixo. Com isto são necessários alguns cálculos auxiliares de forma a obter uma estimativa da posição do centro de massa:

$$m_1 = -\tan\beta_t \tag{4.18}$$

$$b_t = -\left(\frac{a_{\beta_t}}{\cos\beta_t} + r_R\right) \tag{4.19}$$

As distâncias horizontais αB e os ângulos β :

$$a_B = \begin{bmatrix} 0.101 \\ 0.256 \\ 0.299 \end{bmatrix} \text{m} \tag{4.20}$$

$$\beta_B = \begin{bmatrix} 197^o \\ 103^o \\ 328^o \end{bmatrix} \tag{4.21}$$

O centro de massa do quadro pode ser então obtido intersetando as três linhas representadas na *Figura 4.1*. Isto é realizado por meio do cálculo da intersecção entre duas linhas. As equações para a intersecção entre a linha 1 e 2 estão descritas abaixo, *equação 4.22*, enquanto as equações para os outros cruzamentos de linha podem ser facilmente derivadas das equações para a linha 1 e 2.

$$\begin{bmatrix} -m_1 & 1 \\ -m_2 & 1 \end{bmatrix} \begin{bmatrix} x_a \\ z_a \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{4.22}$$

Os três cruzamentos efetuam a média de forma a obter uma posição mais precisa param o centro de massa:

$$x_B = \frac{x_a + x_b + x_c}{3} \tag{4.23}$$

$$z_B = \frac{z_a + z_b + z_c}{3} \tag{4.24}$$

O centro de massa é de seguida descrito pelo seguinte:

$$cm_B = [x_B \quad 0 \quad z_B] \tag{4.25}$$

$$cm_B = [-0.1350 \quad 0 \quad -0.4653] \tag{4.26}$$

4.6.3 Forqueta e Guiador

O centro de massa para o conjunto da forqueta e guiador foi encontrado de uma forma semelhante à efetuada anteriormente para o quadro, utilizando o pêndulo de torção. Assim, este conjunto foi pendurado em duas direções diferentes, primeiro com o eixo de direção na horizontal e, em seguida, com o eixo na vertical. Em ambos os sentidos considerou-se o centro de massa na extensão do braço do pêndulo. A distância horizontal αH e os ângulos β foram medidos:

$$a_H = \begin{bmatrix} 0.324 \\ -0.006 \end{bmatrix} \tag{4.27}$$

$$\beta_H = \begin{bmatrix} 22^o \\ 295^o \end{bmatrix} \tag{4.28}$$

A interceção com o eixo z usada para calcular o centro de massa da forqueta e guiador é alterada para:

$$b_t = w \tan(\beta_t) - r_F - \left(\frac{aH_t}{\cos \beta_t}\right) \tag{4.29}$$

O centro de massa é encontrado a partir de duas linhas, resolvendo para isso a *equação 4.22*. O centro de massa é dado então pela seguinte matriz:

$$cm_H = [x_a \quad 0 \quad z_a] \tag{4.30}$$

$$cm_H = [0.6181 \quad 0 \quad -0.4811] \tag{5.31}$$

4.7 Momento de Inércia

O momento de inércia foi medido utilizando o pressuposto de que a bicicleta é simétrica sobre o plano XZ. O momento de inércia em torno dos eixos X e Z foi medido utilizando um pêndulo de torção, enquanto o momento da inércia em torno do eixo Y foi obtido utilizando para tal efeito um pêndulo composto. As oscilações foram medidas usando um cronômetro comum com uma resolução de 1/100 de segundo. O tempo foi medido num período entre 10 a 20 oscilações, dependendo da frequência, e cada medição, foi realizada três vezes. Todos os cálculos foram efetuados utilizando uma rotina no *MATLAB* que pode ser encontrada no *Apêndice B.2*, sendo esta uma adaptação para este protótipo, da rotina descrita no trabalho referenciado por [8].

4.7.1 Pêndulo de Torção

Um pêndulo de torção foi criado para a medição do momento de inércia em torno dos eixos X e Z. O pêndulo de torção consiste em três partes diferentes, uma cabo de torção, uma braçadeira superior usada para prender o cabo de torção ao teto e uma variação de combinações de sistemas de aperto para prender a bicicleta ao cabo de torção. O sistema de aperto é articulado usando para tal diferentes abraçadeiras de nylon, prendendo assim as diferentes partes da bicicleta nos ângulos pretendidos. Uma barra de ferro foi ligada ao pêndulo de forma a ser controlada a sua posição horizontal, confirmando-se essa, usando um nível de bolha e ajustando os parafusos de aperto à barra. Assume-se a precisão dos ângulos de ± 1º. O pêndulo de torção foi calibrado usando a barra de ferro descrita com um momento de inércia conhecido, cujo é calculado pela *equação 4.32*, e irá servir de apoio para prender as diferentes partes da bicicleta.

$$I_{calib} = \frac{m_{calib}}{12}\left(3{r_{calib}}^2 + {L_{calib}}^2\right) \tag{4.32}$$

Os tempos medidos 10 oscilações da haste de calibração:

$$t_{calib} = \begin{bmatrix} 14.95 \\ 15.61 \\ 14.30 \end{bmatrix} \text{s} \tag{4.33}$$

O período médio de oscilação pode ser calculado pela seguinte equação:

$$\bar{T} = \frac{t_1 + t_2 + t_3}{3v} \tag{4.34}$$

Onde v corresponde ao número de oscilações, juntamente com o momento de inércia conhecido para a haste de calibração, a rigidez do pêndulo de torção pode ser estimada pela seguinte equação:

$$k = \frac{4I_{calib}\pi^2}{{\bar{T}_{calib}}^2} \tag{4.35}$$

Usando o pêndulo de torção o momento de inércia em torno do pêndulo eixo pode ser encontrado através da seguinte equação:

$$J = \frac{4\bar{T}^2}{4\pi^2} \tag{4.36}$$

4.7.2 Pêndulo Composto

De forma a medir o momento de inércia em relação ao eixo Y utiliza-se o princípio do pêndulo composto. Para as rodas usou-se então duas pequenas guias com um orifício central, montadas entre duas cadeiras, onde sobre elas se encontra apoiada uma haste presa na jante interna da roda, podendo movimentar-se praticamente sem atrito. O momento de inércia pode ser então calculado a partir do momento de oscilação com a *equação 4.37*, em que m é a massa e L é o comprimento desde o ponto de rotação até ao centro de massa da parte medida.

$$I_{yy} = \left(\frac{\bar{T}}{2\pi}\right)^2 mgI_c - mI_c{}^2 \tag{4.37}$$

4.7.3 Rodas

Para a determinação do momento de inércia das rodas assumiu-se que estas são simétricas sobre os três planos ortogonais. O momento de inércia em torno dos eixos X e Z é idêntico e foi medido pendurando a roda no pêndulo de torção. Já o momento de inércia em torno do eixo Y foi medido suspendendo as rodas utilizando o pêndulo composto. A torção da roda da frente (a) e da roda traseira (b) e as medições do pêndulo composto foram realizadas medindo o tempo para 20 oscilações:

(a) Roda Traseira (b) Roda Dianteira

Figura 4.3 - Pêndulo de Torção com as rodas para medição;

$$t_{Fxz} = \begin{bmatrix} 18.87 \\ 19.02 \\ 19.13 \end{bmatrix} \text{s} \tag{4.38}$$

$$T_{Rxz} = \begin{bmatrix} 19.46 \\ 19.91 \\ 19.52 \end{bmatrix} \text{s} \tag{4.39}$$

(a) Roda Traseira (b) Roda Dianteira

Figura 4.4 - Pêndulo Composto;

$$t_{Fy} = \begin{bmatrix} 20.99 \\ 20.61 \\ 20.72 \end{bmatrix} \text{s} \tag{4.40}$$

$$t_{Ry} = \begin{bmatrix} 19.70 \\ 19.65 \\ 19.53 \end{bmatrix} \text{s} \tag{4.41}$$

O momento de inércia para os eixos X e Z pode ser calculado utilizando as *equações 4.34 e 4.36*. Enquanto que o momento de inércia para o eixo Y pode ser calculado com o recurso às *equações 4.34 e 4.37* e com o comprimento do pêndulo *lcfw = lcrw = 0.136 m*, a distância medida a partir do eixo de rotação do pêndulo composto até ao centro da roda. O momento de inércia para a roda da frente *IF* e a para a roda traseira *IR:*

$$I_F = [0.0123 \quad 0.0204 \quad 0.0123] \tag{4.42}$$

$$I_R = [0.0131 \quad 0.0188 \quad 0.0131] \tag{4.43}$$

4.7.4 Quadro

Para a estrutura do corpo traseiro, quadro e componentes, apenas é necessário o momento de inércia em torno dos eixos X e Z. Este foi então medido recorrendo ao pêndulo de torção, onde o quadro foi pendurado em três ângulos diferentes, sendo estes perpendiculares ao topo, assento e tubos de trás. As medições foram realizadas medindo o tempo para v = 20 oscilações e os ângulos β são os mesmos que foram utilizados para o cálculo do centro de massa.

Figura 4.5 - Medidas efetuadas ao quadro;

$$t_{Bst} = \begin{bmatrix} 41.75 \\ 42.31 \\ 42.10 \end{bmatrix} \text{s} \tag{4.44}$$

$$t_{Btt} = \begin{bmatrix} 58.73 \\ 58.61 \\ 58.82 \end{bmatrix} \text{s} \tag{4.45}$$

$$t_{Bbt} = \begin{bmatrix} 58.65 \\ 56.32 \\ 57.25 \end{bmatrix} \text{s} \tag{4.46}$$

O momento de inércia em torno dos eixos do pêndulo pode ser calculado pelas *equações 4.34 e 4.36*. O momento de inércia sobre os eixos X e Z pode ser calculado através da formulação de uma relação entre os quadros de inércia:

$$J_t = R_i I {R_i}^T \tag{4.47}$$

Onde *Ji* é o momento de inércia em torno dos eixos de oscilação. *I* é o momento de inércia sobre o plano de referência e *Ri* é a matriz rotacional. As matrizes *I* e *Ri* são reduzidas a matrizes 2 × 2 uma vez que o eixo Y dos dois quadros são os mesmos assumindo-se assim simetria lateral:

$$I = \begin{bmatrix} I_{xx} & I_{xz} \\ I_{zx} & I_{zz} \end{bmatrix} \tag{4.48}$$

$$R_i = \begin{bmatrix} \cos\beta_t & -\sin\beta_t \\ \sin\beta_t & \cos\beta_t \end{bmatrix} \tag{4.49}$$

Ji pode ser calculado, mas apenas a primeira linha é necessária:

$$J_i = (\cos\beta_t I_{xx})^2 - 2\sin\beta_t \cos\beta_t I_{xz} + (\sin\beta_t I_{zz})^2 \tag{4.50}$$

Isso permite formar:

$$\begin{bmatrix} J_1 \\ J_2 \\ J_3 \end{bmatrix} = \begin{bmatrix} (\cos\beta_1)^2 & -2\sin\beta_1\cos\beta_1 & (\sin\beta_1)^2 \\ (\cos\beta_2)^2 & -2\sin\beta_2\cos\beta_2 & (\sin\beta_2)^2 \\ (\cos\beta_3)^2 & -2\sin\beta_3\cos\beta_3 & (\sin\beta_3)^2 \end{bmatrix} \begin{bmatrix} I_{xx} \\ I_{xz} \\ I_{zz} \end{bmatrix} \tag{4.51}$$

O momento de inércia pode ser descrito pela seguinte matriz:

$$I = \begin{bmatrix} I_{xx} & 0 & I_{xz} \\ 0 & I_{yy} & 0 \\ I_{zx} & 0 & I_{zz} \end{bmatrix} \tag{4.52}$$

A inércia para a estrutura do quadro da bicicleta, *Iyy* é ajustado para zero, uma vez que não é usado pelas equações linearizadas:

$$I_B = \begin{bmatrix} 0.0753 & 0 & 0.0317 \\ 0 & 0 & 0 \\ 0.0317 & 0 & 0.1049 \end{bmatrix} \tag{4.53}$$

4.7.5 Forqueta e Guiador

O momento de inércia para o conjunto da forqueta e guiador é encontrado utilizando o pêndulo de torção de forma a obter o momento de inércia sobre os eixos X e Z. As medições de torção e os cálculos são praticamente os mesmos dos efetuados para a roda da frente, agora com os seguintes ângulos *βH*, tempos *tH* e o número de oscilações v_H;

Figura 4.6 - Medidas efetuadas ao conjunto guiador e forqueta;

$$v_H = \begin{bmatrix} 10 \\ 20 \end{bmatrix} \tag{4.54}$$

$$\beta_H = \begin{bmatrix} 22^o \\ 295^o \\ 153^o \end{bmatrix} \tag{4.55}$$

$$t_{H1} = \begin{bmatrix} 25.13 \\ 25.31 \\ 25.46 \end{bmatrix} \text{s} \tag{4.56}$$

$$t_{H2} = \begin{bmatrix} 17.09 \\ 18.37 \\ 18.20 \end{bmatrix} \text{s} \tag{4.57}$$

$$t_{H3} = \begin{bmatrix} 20.12 \\ 20.40 \\ 20.23 \end{bmatrix} \text{s} \tag{4.58}$$

O momento de inércia para o conjunto frontal pode ser escrito na forma vista na *equação 4.52*. Onde *IHyy* é definido como zero, uma vez que não é utilizado nas equações linearizadas:

$$I_H = \begin{bmatrix} 0.0753 & 0 & -0.0199 \\ 0 & 0 & 0 \\ -0.0199 & 0 & 0.0549 \end{bmatrix} \tag{4.59}$$

Capítulo 5

Simulações

Neste capítulo irão ser apresentadas as simulações efetuadas para o sistema físico, composto pela bicicleta e os respetivos elementos adicionais que o constituem. Esta análise irá visar a composição dos modelos e explicitar os detalhes das diversas simulações realizadas, tendo por base situações que possam vir a acontecer em tempo real. A obtenção dos parâmetros, para o modelo de bicicleta *Whipple* [10], foi realizada da forma já explicada no *Capítulo 4*.

5.1 Ambientes de Simulação

Esta secção irá descrever os ambientes de simulação utilizados em *Simulink*, compostos pelo modelo da bicicleta criado, sinais de teste e ruído, e blocos adicionais que sejam úteis à simulação genérica. De referir que em todas as simulações efetuadas neste *ponto 5.1*, foi utilizado um valor para a amostragem de *h=0.01*, correspondendo a uma frequência de 100 Hz.

5.1.1 Modelo da Bicicleta

O modelo de bicicleta *Whipple* [10] descrito no *Capítulo 2* foi modelado utilizando os parâmetros encontrados no *Capítulo 4,* após respetivas medições efetuadas. Os parâmetros medidos, estimados e calculados neste capítulo, podem ser vistos em mais detalhe no *Apêndice B.3*. Para tal, as equações necessárias a essa resolução foram calculadas com a ajuda do *MATLAB* [1], utilizando um *m-file* escrito para o propósito. Esse código pode ser encontrado no *Apêndice B.4*, sendo esta uma adaptação para este protótipo, da rotina descrita no trabalho referenciado por [8]. O *m-file* calcula as 4 matrizes abaixo, necessárias para completar o modelo descrito pela *equação 2.2*.

$$\mathbf{M} = \begin{bmatrix} 2.3552 & 0.0740 \\ 0.0740 & 0.1049 \end{bmatrix} \tag{5.1a}$$

$$\boldsymbol{K0} = \begin{bmatrix} -4.9348 & -0.1543 \\ -0.1543 & -0.0795 \end{bmatrix} \tag{5.1b}$$

$$\boldsymbol{K2} = \begin{bmatrix} 0 & 5.8323 \\ 0 & 0.2368 \end{bmatrix} \tag{5.1c}$$

$$\mathbf{C1} = \begin{bmatrix} 0 & 0.9403 \\ -0.1168 & 0.3465 \end{bmatrix} \tag{5.1d}$$

A *equação 2.2* pode ser reorganizada na forma:

$$\ddot{q} = M^{-1}(f - vC_1\dot{q} - [gK_0 + v^2K_2]q) \tag{5.2}$$

A partir desta equação é possível criar o modelo da bicicleta no *Simulink*, tal como pode ser visto na *Figura 5.1.* Tendo em conta as noções já apresentadas no *Capítulo 2* a identificação dos blocos do sistema correspondentes torna-se elementar.

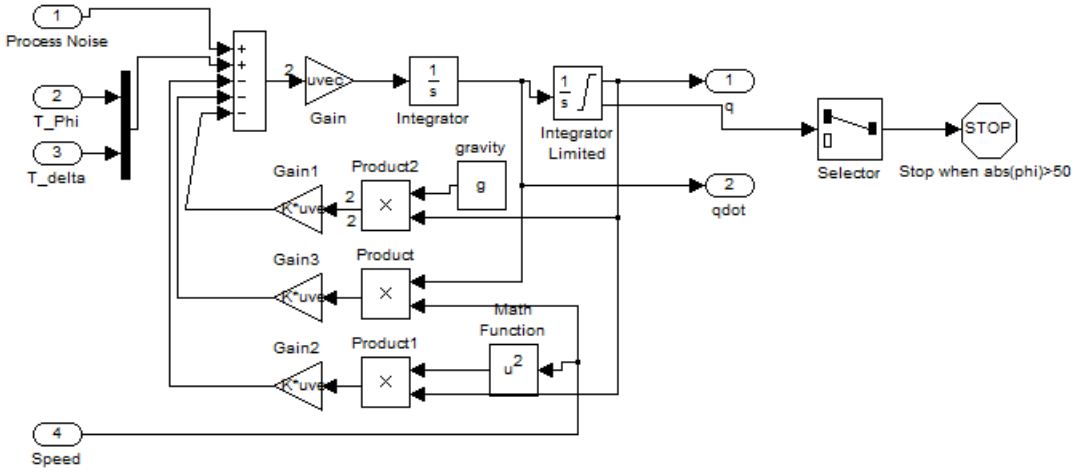


Figura 5.1 - Diagrama em *Simulink* do modelo da Bicicleta descrito pela equação 5.2;

Foi adicionado um bloco integrador com limitador, de forma ao sistema parar a simulação quando a inclinação atinge um determinado valor. Neste caso, quando o módulo da inclinação for maior que 1 radiano, ou seja, aproximadamente 57 graus.

5.1.2 Simulação do Modelo da Bicicleta

Por forma a testar o modelo criado na *secção 5.1.1* foi criado um ambiente de simulação para o efeito. Primeiramente, como pode ser visto na *Figura 5.2*, considera-se apenas o modelo em si, sinais de entrada e saída, e uma atribuição aleatória de ruído. Este último é adicionado aos canais pelo porto de entrada *Process Noise* e no caso destas simulações, assume as seguintes características:

$$Média\ do\ ruído\colon\ \bar{r} = 0; \tag{5.3a}$$

$$Variância\ do\ ruído\colon\ \partial = 0.2\ rad; \tag{5.3b}$$

Este ruído pretende simular as perturbações naturais do movimento de uma bicicleta como, por exemplo, irregularidades do piso. Foi adicionado de igual modo, um bloco que pretende simular uma perturbação externa ao sistema, representada pelo *Signal Builder*, devolvendo assim uma força exercida, que impõe uma inclinação expressa em radianos. Assim é possível simular o modelo da bicicleta quando a este não lhe é aplicado qualquer sinal de controlo, obtendo para além de outras saídas, a inclinação da bicicleta e os ângulos de rotação do volante, expressos por *Lean_Phi* e *Steer_deltha* respetivamente.

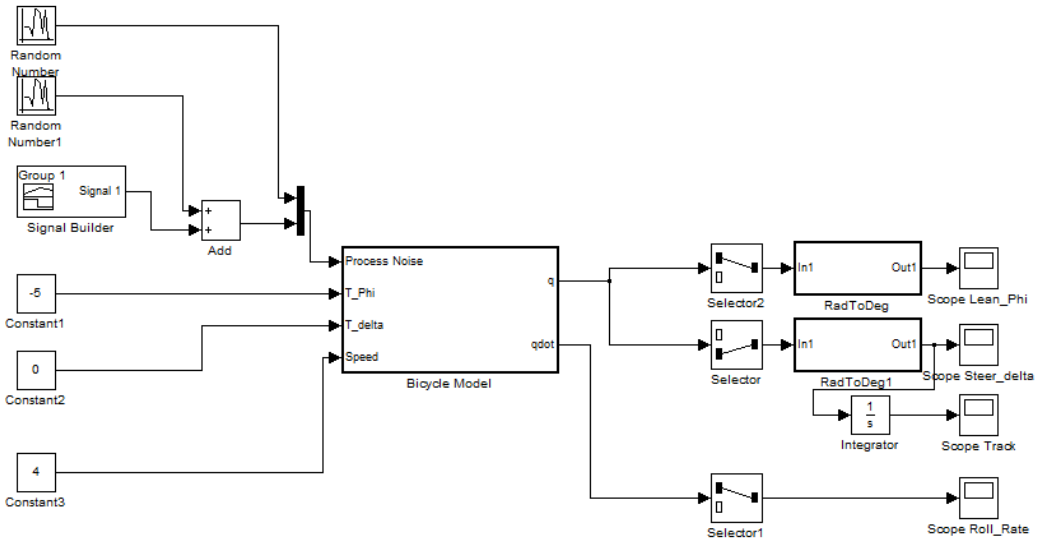


Figura 5.2 - Ambiente de Simulação da Bicicleta sem controlo;

Os blocos *Constant1* e *Constant2* representam os *Setpoints* iniciais que o modelo da bicicleta deve considerar durante a simulação. Estes encontram-se expressos em radianos. O bloco *Constant3* representa a velocidade constante da bicicleta descrita em metros por segundo. O controlo desta não será aqui simulado, considerando assim que o controlador de velocidade externo funciona corretamente. Foi igualmente criado um bloco que se adicionou a todos os ambientes de simulação, cuja função é converter a saída dada em radianos, em graus, para uma melhor perceção da inclinação e direção.

5.1.2.1 Inclinação de 0.5 radianos

Ao assumir que a bicicleta já se encontra inclinada com um valor de 0.5 radianos no início da simulação (correspondentes a aproximadamente 28.6º em *T_Phi*), é possível ver a evolução no tempo da mesma inclinação, num cenário de teste com uma duração de 30 segundos, *Figura 5.3*. A velocidade de 4 m/s foi atribuída num contexto genérico. Identifica-se facilmente que a bicicleta cai num curto espaço de tempo (medição do

tempo no eixo do *X's*), no sentido positivo do eixo responsável pela medição da inclinação final da bicicleta *Y's*, dado nesta *Figura 5.3* em graus.

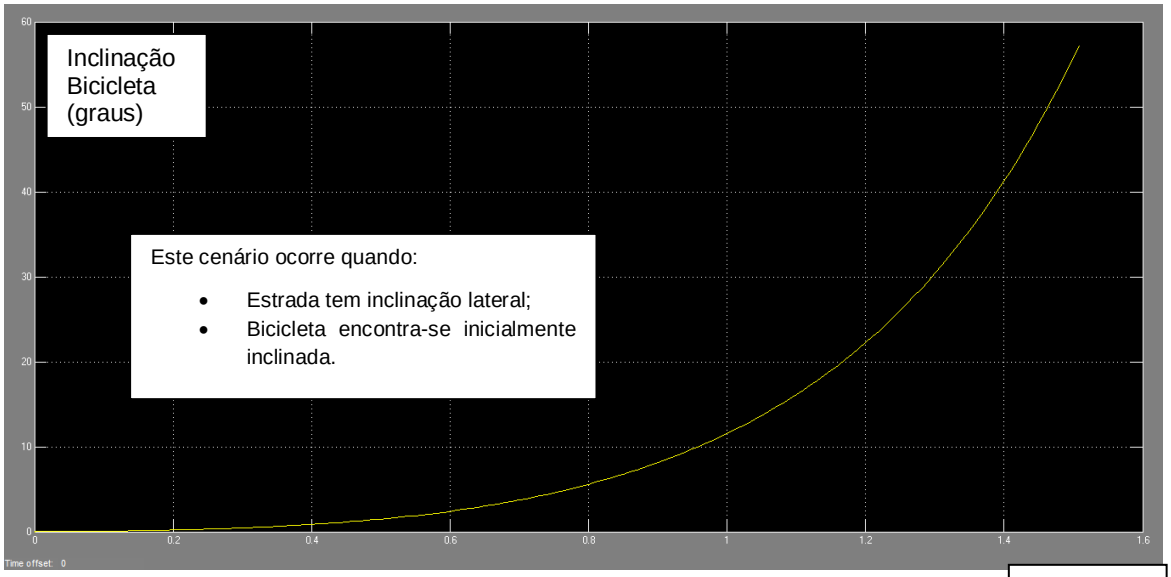


Figura 5.3 - Inclinação Inicial -0.5 radianos;

5.1.2.2 Inclinação de -0.5 radianos

Ao assumir que a bicicleta se encontra num local que impõe uma inclinação de valor de -0.5 radianos, ou seja, o cenário oposto ao ponto anterior identifica-se facilmente que a bicicleta apresenta um comportamento simétrico caindo igualmente num curto espaço de tempo, *Figura 5.4*, mas desta vez no sentido negativo do eixo dos graus de inclinação.

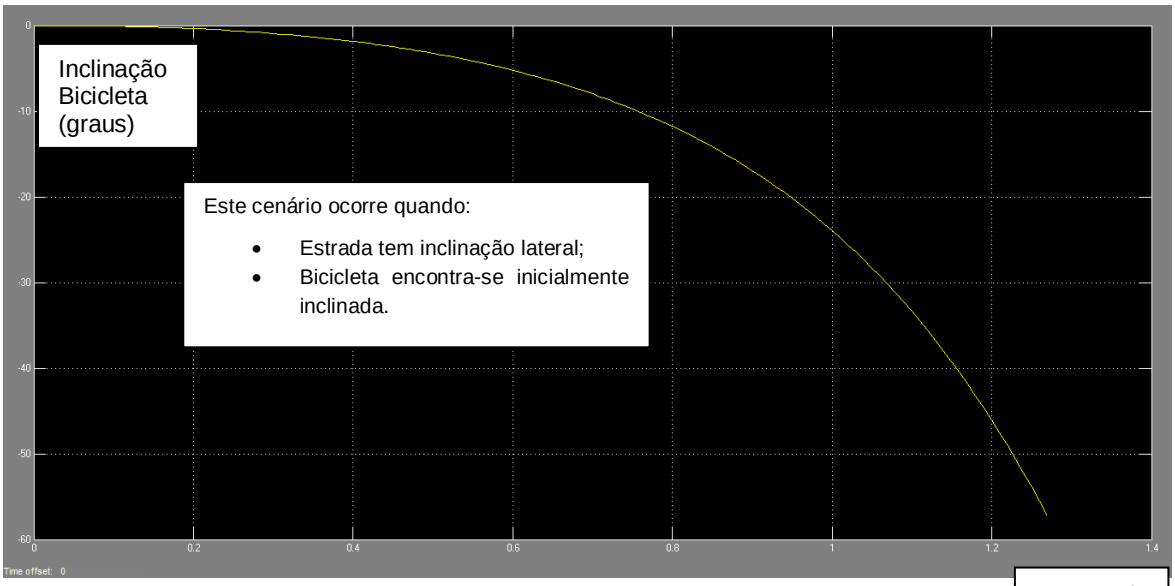


Figura 5.4 - Inclinação Inicial -0.5 radianos;

5.2 Simulações Controladas

Nesta secção irão ser apresentadas soluções de controlo de equilíbrio do modelo da bicicleta, apresentando dois tipos de controlador para a função pretendida, explicitando-os em detalhe. De igual modo serão apresentadas as simulações efetuadas, já com a estimação dos parâmetros de controlo capazes de o realizar, para o passo de amostragem escolhido.

5.2.1 Escolha do Passo de Amostragem

Uma das escolhas iniciais a ser efetuada trata-se da escolha de um passo de amostragem adequado à controlabilidade do sistema. Este aspeto torna-se mais simples de ser analisado se for aqui descrito em frequência. De grosso modo poder-se-ia assumir uma frequência de 10 Hz (*h = 0.1s*), onde o algoritmo de controlo efetuaria dez ciclos a cada segundo, e já que a bicicleta demora perto de 1 segundo a atingir um ponto critico, este passo de amostragem parece, do ponto de vista de simulação, suficiente. Do ponto de vista de simulação irão ser apresentados alguns parâmetros capazes de neste ambiente de *Simulink* controlar o modelo da bicicleta para um conjunto de passos de amostragem diferentes (10 Hz, 100 Hz, 500 Hz). No entanto, como irá ser descrito no *Capítulo 6,* no ambiente de teste real torna-se impossível controlar a bicicleta utilizando algumas gamas de passos de amostragem, devido aos tempos de atuação dos diferentes módulos envolvidos. Assim optou-se pela frequência de 100 Hz *(h = 0.01s)*, tanto nos testes reais, bem como nas simulações para possível análise e comparação.

5.2.2 Controlador PID

O primeiro controlador a ser simulado, trata-se de um *Proporcional Integral Derivative*, PID [15], que do ponto de vista de controlabilidade e robustez parece à priori, ser o mais indicado para resolver este problema de controlo de equilíbrio. Fácil de compreender e aplicar num microcontrolador, gera algoritmos rápidos e eficientes para sistemas pouco complexos. A componente proporcional produz um sinal de saída que é proporcional à amplitude do erro. Uma vez que o sinal de saída é proporcional ao erro, um erro não-nulo, *offset*, é gerado. Um ganho proporcional muito alto pode destabilizar o sistema. A componente integral produz um sinal de saída que é proporcional à magnitude e à duração do erro, ou seja, ao erro acumulado. Isto proporciona uma solução para corrigir o erro de *offset* criado pela componente proporcional. Se o ganho integral for baixo, o sistema pode levar muito tempo a atingir o valor de referência. No entanto, se o ganho integral for muito alto, o sistema pode tornar-se instável. Por fim a componente derivativa produz um sinal de saída que é proporcional à velocidade de variação do erro. Esta fornece uma correção antecipada do erro, diminuindo o tempo de resposta e melhorando a estabilidade do sistema. A *Figura 5.5* representa o ambiente de simulação criado para testar a aplicação deste controlador.

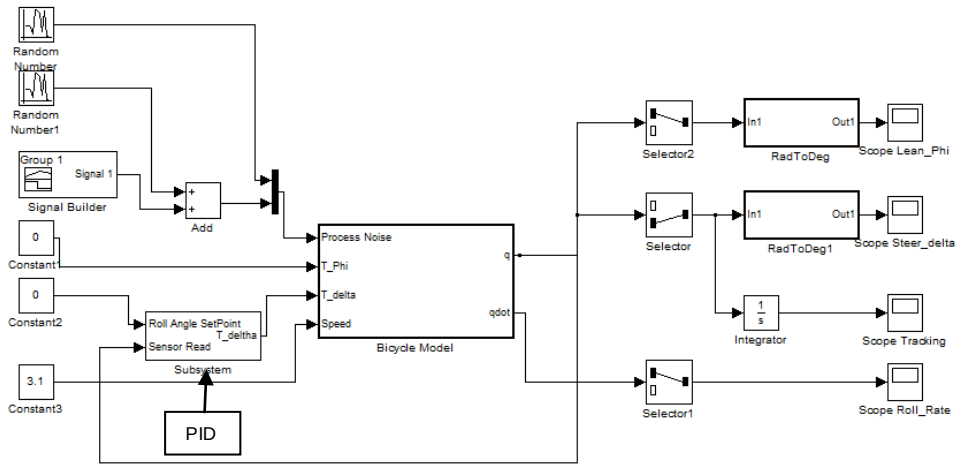


Figura 5.5 - Ambiente de Simulação da Bicicleta com controlador PID;

5.2.2.1 Bloco do Controlador PID e Parâmetros

Ao ambiente de simulação da *Figura 5.2* foi então adicionado um bloco de controlo, que execute o algoritmo do controlador PID. Este recebe os valores lidos do sensor, nomeadamente a inclinação e o estado do volante, e após a execução deste algoritmo, devolve um valor de rotação da direção, em radianos. O bloco foi construído e está representado na *Figura 5.6*.

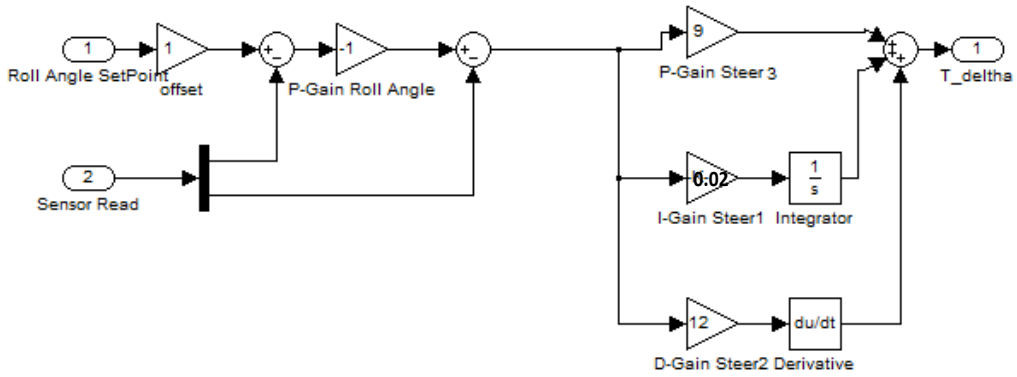


Figura 5.6 - Bloco do Controlador PID;

Os valores dos parâmetros apresentados na *Figura 5.6,* são valores que pretendem representar os parâmetros indicados para o correto funcionamento do sistema. Estes serão utilizados em todas as simulações apresentadas de modo a unificar as mesmas, usando cenários de teste idênticos. De referir que se trata de parâmetros aproximados e servem apenas de base para a programação no sistema físico, podendo estes variar aquando da aplicação ao mundo real. São estes, *Kp = 9, Ki = 0.02, Kd = 12.* Assemelha-se muito a um controlador do tipo PD, como seria o esperado, uma vez que é um sistema

que necessita de uma capacidade elevada de previsão, concedida pelo elemento derivativo. A velocidade com que irão ser efetuadas as simulações corresponde a uma velocidade superior a 3 m/s e inferior a 5 m/s, sendo uma gama de valores capazes de serem produzidos pelo motor DC, adicionado à bicicleta, e controlados por um controlador adicional. Em ambiente de simulação considera-se aleatoriamente, *v = 3.1 m/s,* de forma a recriar uma certa gama de incerteza em torno dos 3 m/s, visto se tratar da velocidade limite de controlabilidade, ou seja, velocidades inferiores a este valor tornam o sistema praticamente incontrolável utilizando um algoritmo de controlo PID. No mundo real o sistema de velocidade irá ser controlado a uma velocidade de 3.8 m/s, não retirando valor a estas simulações, já que se tratam de um ponto de partida para o controlo de equilíbrio do modelo real, sendo estes resultados semelhantes para as duas velocidades.

5.2.2.2 Simulação sem Perturbações externas

Assumindo um cenário de teste onde o *Signal Builder* esteja desligado, ou seja, não existe uma perturbação externa aplicada ao sistema, apenas o ruído criado por atrito, efetua-se uma simulação durante um período de 120 segundos. Na *Figura 5.7* está apresentada uma das saídas da simulação controlada por um controlador PID, tratando-se da inclinação da bicicleta, medida em graus, em função do tempo.

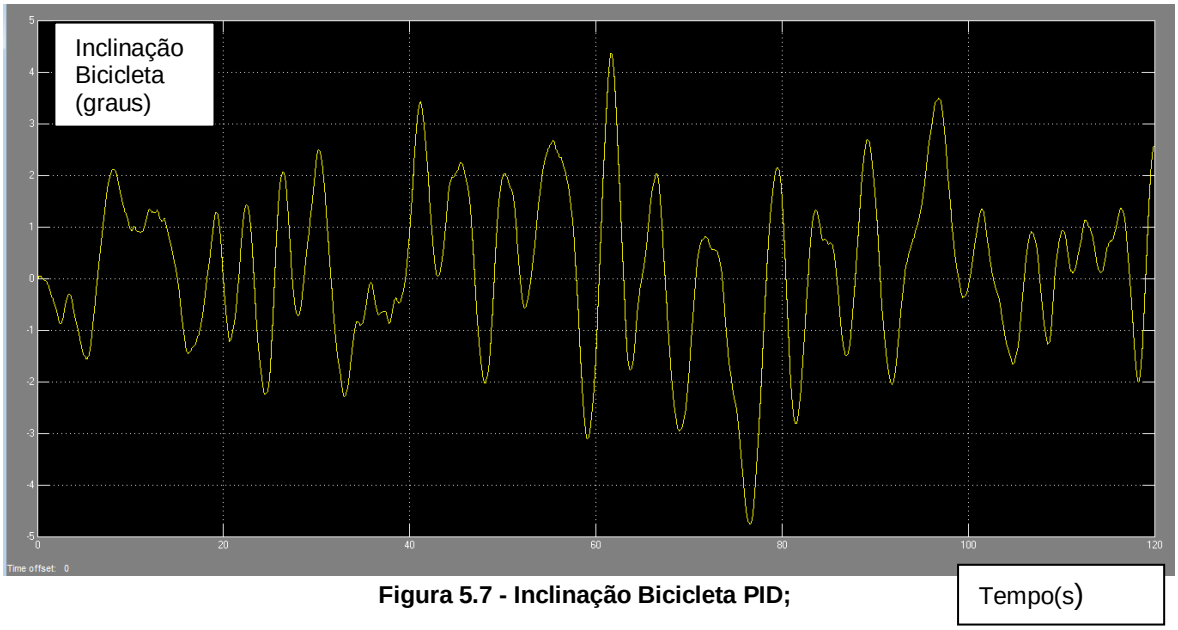


Figura 5.7 - Inclinação Bicicleta PID;

É possível verificar que o algoritmo funciona perfeitamente para os parâmetros escolhidos. Estes são valores aceitáveis, gerando assim um sinal de controlo de ordem adequada ao sistema. Trata-se de uma resposta que mantem a inclinação da bicicleta dentro de ângulos fisicamente aceitáveis e realizáveis através de um servo-motor, equilibrando-a. Para além da inclinação é possível analisar de igual modo na *Figura 5.8*

os ângulos que a direção efetua. Na prática, este angulo que a direção efetua necessita de ser convertido para o angulo respetivo do servomotor, podendo assim variar ligeiramente os parâmetros aqui encontrados.

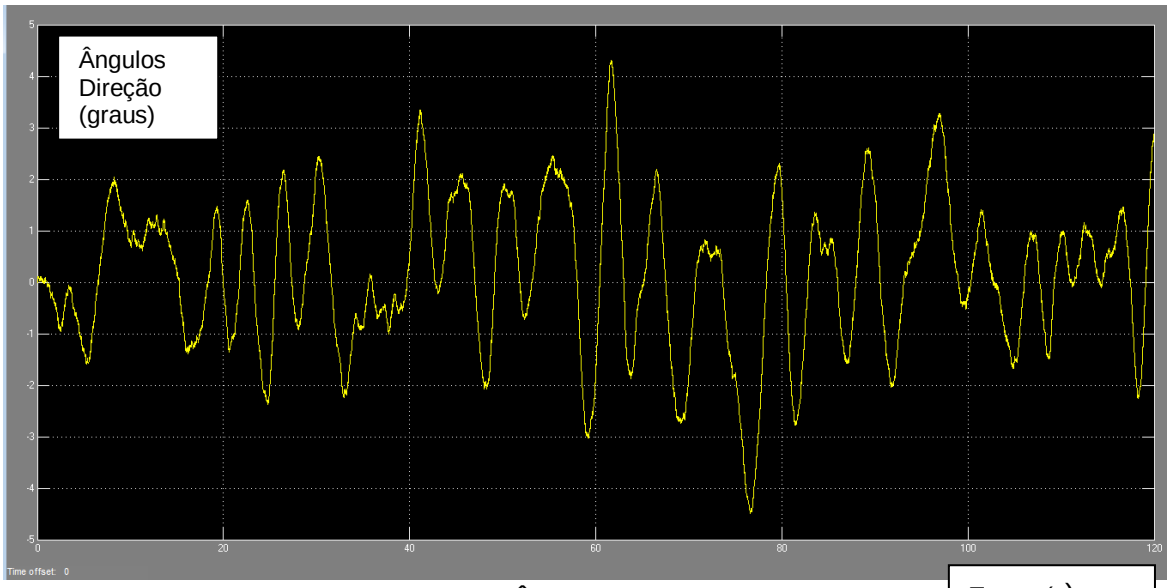


Figura 5.8 - Ângulos da direção PID;

Na *Figura 5.9* é apresentado aquilo que pode ser considerado por o deslocamento em metros que a bicicleta percorre ao longo do tempo, para a velocidade fixa escolhida. Este gráfico pode ser analisado como uma vista aérea da posição da bicicleta (eixo dos Y's em metros) no decorrer da simulação (não passando assim de um bloco de integral do angulo de direção da bicicleta).

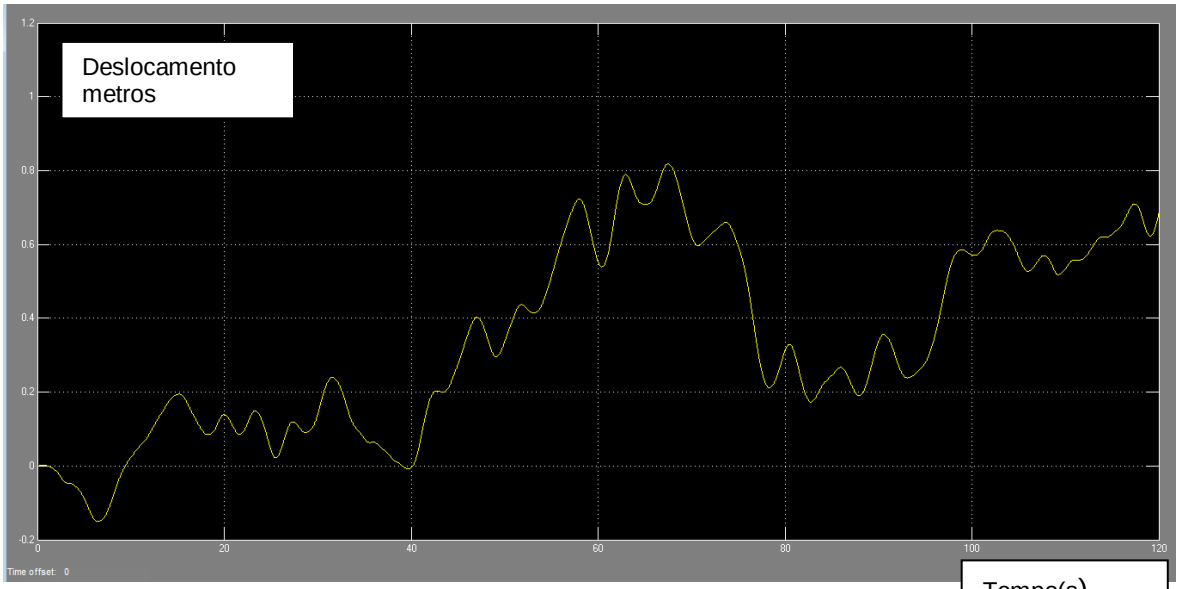


Figura 5.9 - Deslocamento da Bicicleta PID;

5.2.2.3 Simulação com Perturbação Externa a)

Assumindo um cenário de teste onde o *Signal Builder* esteja ativado, ou seja, existe uma perturbação externa aplicada ao sistema, de magnitude -0.25 radianos, efetua-se uma simulação durante um período de 120 segundos. Na *Figura 5.10* está representada essa perturbação, que pode ser atribuída a diversos fatores, como o vento súbito, ou inclinação na estrada.

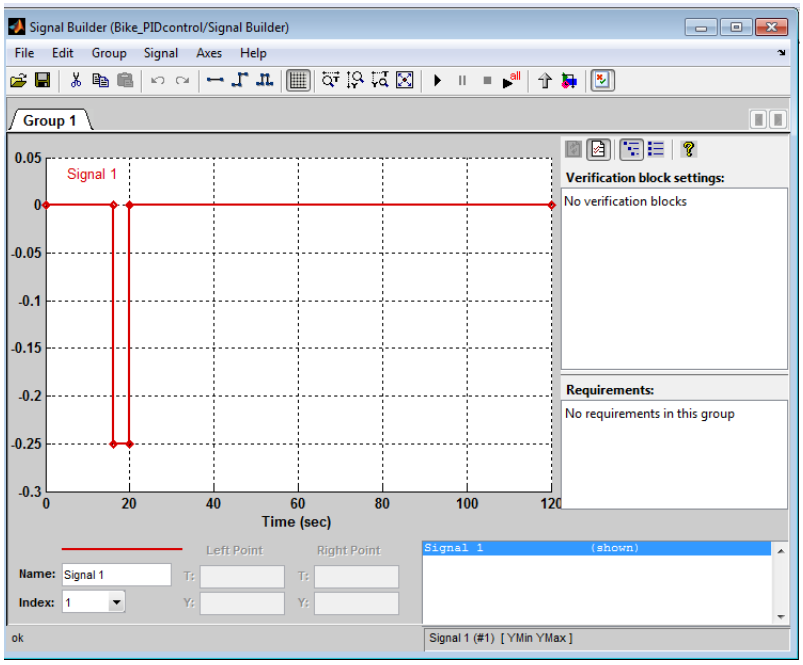


Figura 5.10 - Perturbação Inclinação de -0.25 radianos;

Nas *Figura 5.11* e *Figura 5.12* estão representados dois sinais de saída da simulação controlada por um controlador PID, onde se aplicou uma perturbação externa descrita pela *Figura 5.10*. Essas saídas tratam-se da inclinação da bicicleta, medida em graus, e do "deslocamento" da bicicleta, obtido através da integração dos graus de direção do volante. Os procedimentos e ideais são os mesmos do ponto anterior. Quando se efetua uma análise mais pormenorizada à inclinação produzida neste ensaio, verifica-se facilmente a existência de um pico de "perda" de equilíbrio, notado pela crescente magnitude do sinal, sendo este pico devido à perturbação externa. Do ponto de vista de controlo, o sistema consegue recuperar de perturbações desta magnitude, mantendo a bicicleta dentro da gama de ângulos de equilíbrio. Na *Figura 5.12* é possível compreender o efeito desta perturbação no que toca ao deslocamento ao longo do tempo. Uma inclinação súbita na estrada, ou uma rajada de vento, com magnitude negativa leva o sistema de controlo a compensar esta recorrendo à atuação na direção da bicicleta, resultando assim no deslocamento lateral e metros da mesma.

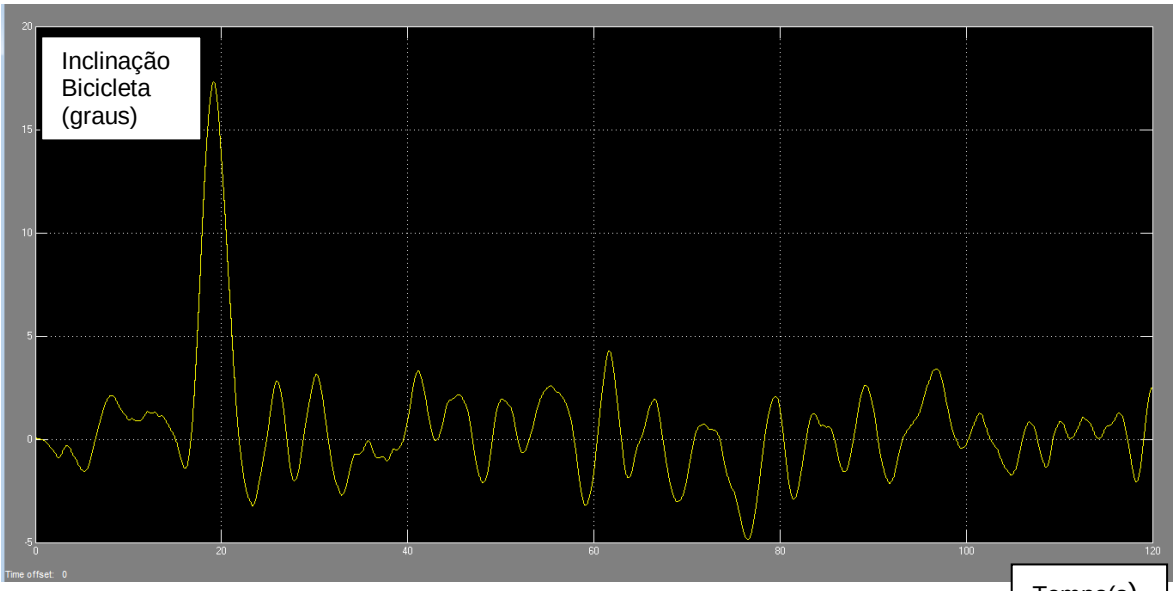


Figura 5.11 - Inclinação da Bicicleta com PID e perturbação -0.25 rad;

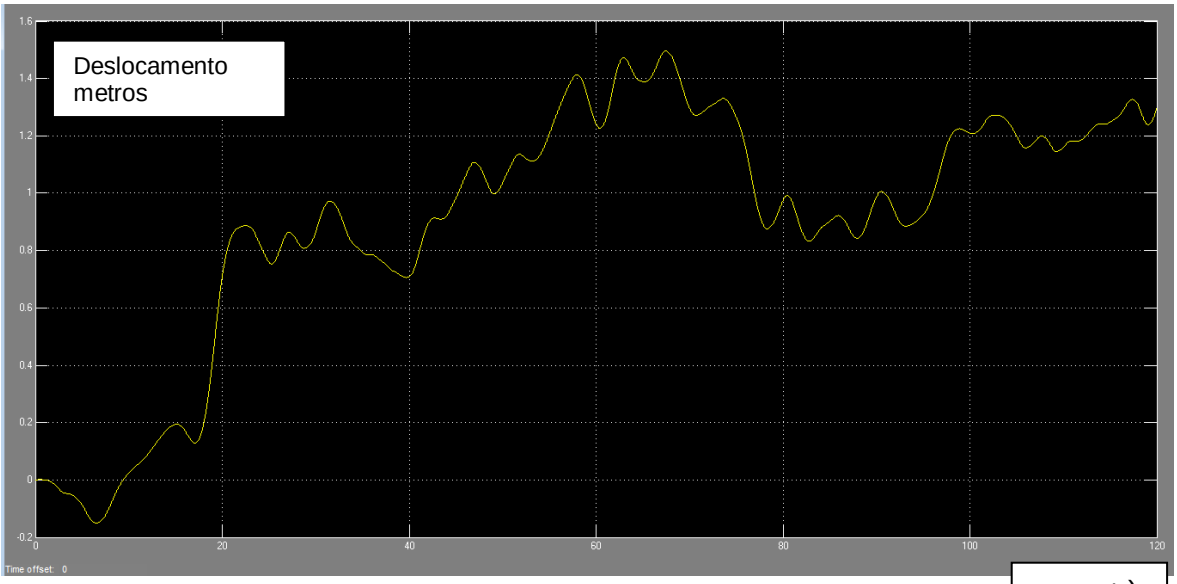


Figura 5.12 - Deslocamento da Bicicleta com PID e perturbação -0.25 rad;

5.2.2.4 Simulação com Perturbação Externa b)

Assumindo um cenário de teste onde o *Signal Builder* esteja ligado, ou seja, existe uma perturbação externa aplicada ao sistema, mas agora de magnitude oposta ao *ponto 5.2.2.3*, ou seja 0.25 radianos, efetua-se uma simulação durante um período de 120 segundos. Nas *Figura 5.13* e *Figura 5.14* estão representadas as saídas mais importantes desta simulação controlada por um controlador PID. Essas saídas, em função do tempo, tratam-se da inclinação da bicicleta, medida em graus e do deslocamento da bicicleta, em metros, obtido através da integração dos graus de direção do volante.

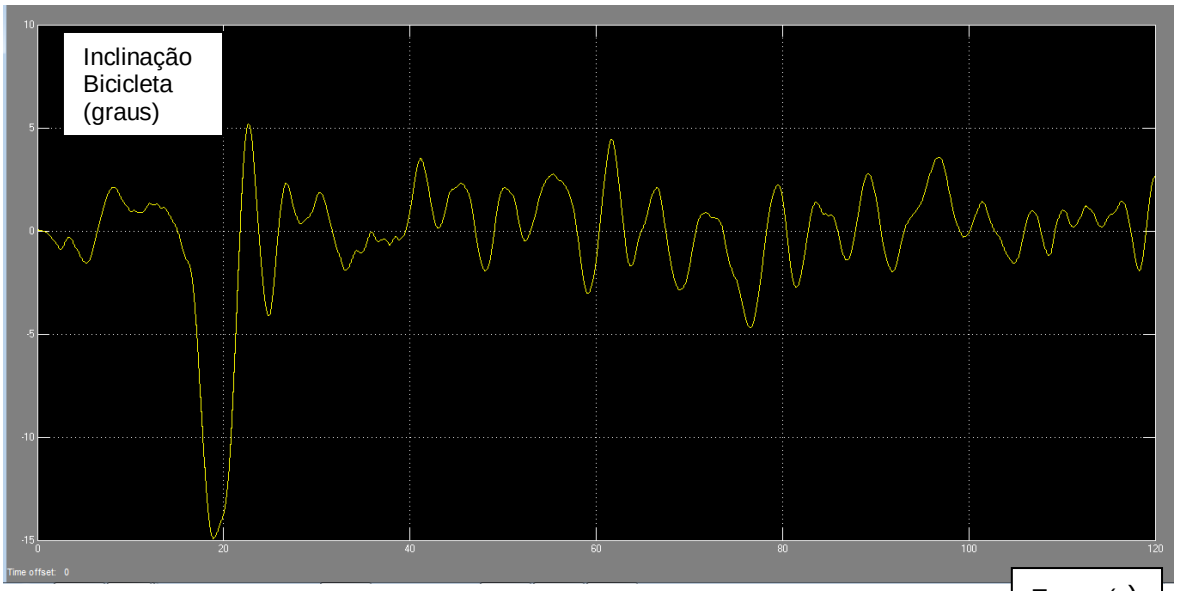


Figura 5.13 - Inclinação da Bicicleta com PID e perturbação 0.25 rad;

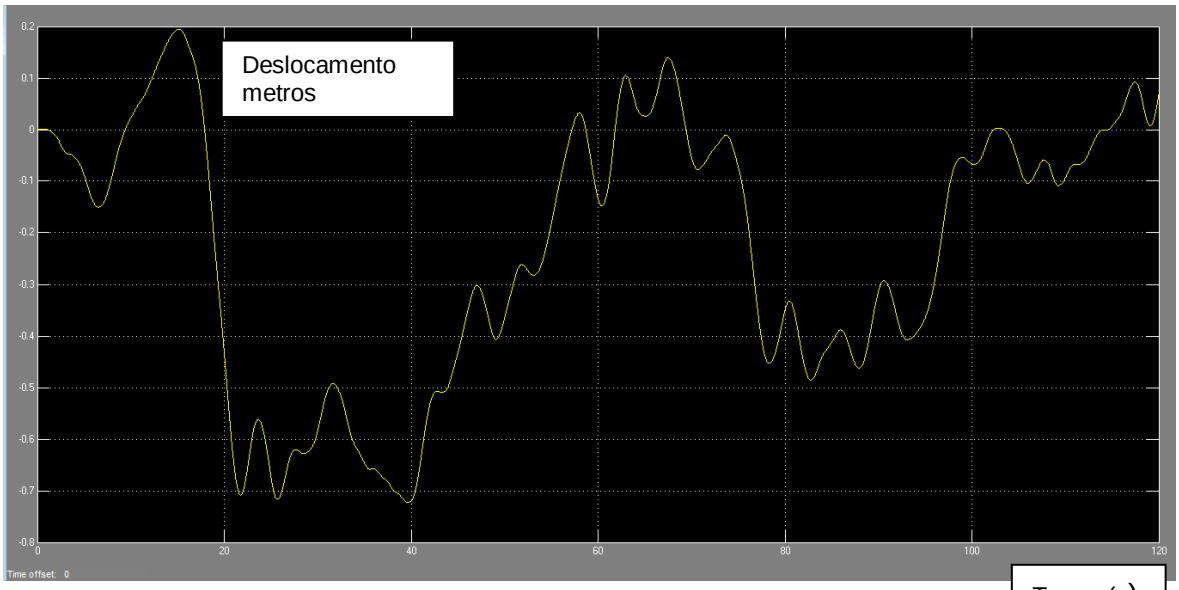


Figura 5.14 - Deslocamento da Bicicleta com PID e perturbação 0.25 rad;

De salientar a simetria apresentada entre as simulação efetuadas nos *pontos 5.2.2.3 e 5.2.2.4*, tanto na inclinação como no deslocamento da bicicleta. Estes resultados confirmam a veracidade do modelo de bicicleta usado, bem como a controlabilidade deste sistema, usando para tal um controlador PID.

5.2.3 Controlador P

Uma vez que o controlador PID simulado no *ponto 5.2.1* apresentou resultados bastante satisfatórios, decidiu-se por curiosidade académica, simular o controlo do sistema usando para tal, um algoritmo mais simples do que o PID. Trata-se assim do uso de um controlador P, proporcional. A *Figura 5.5* representa de igual modo o ambiente de simulação, onde apenas de modificou o bloco do controlador, representado pela *Figura 5.15*.

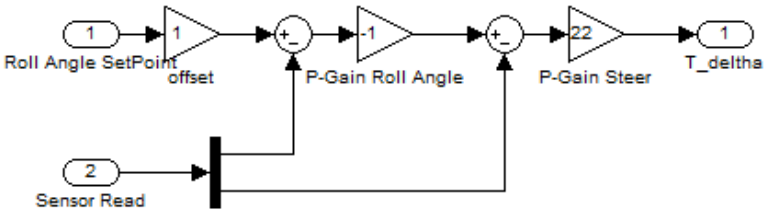


Figura 5.15 - Bloco Controlador Proporcional;

5.2.3.1 Simulações para o Controlador Proporcional

Foi criado um cenário de teste onde o *Signal Builder* proporcionasse uma perturbação externa aplicada ao sistema, de magnitude -0.5 radianos e efetuou-se diversas simulações durante um período de 120 segundos, até o sistema demonstrar controlabilidade, variando para isso o ganho proporcional, Kp. As *Figuras 5.16, 5.17, e 5.18,* representam a variação da inclinação da bicicleta consoante a alteração deste valor. Optou-se também por aumentar a velocidade para 4 m/s, de forma a garantir uma maior hipótese de controlabilidade.

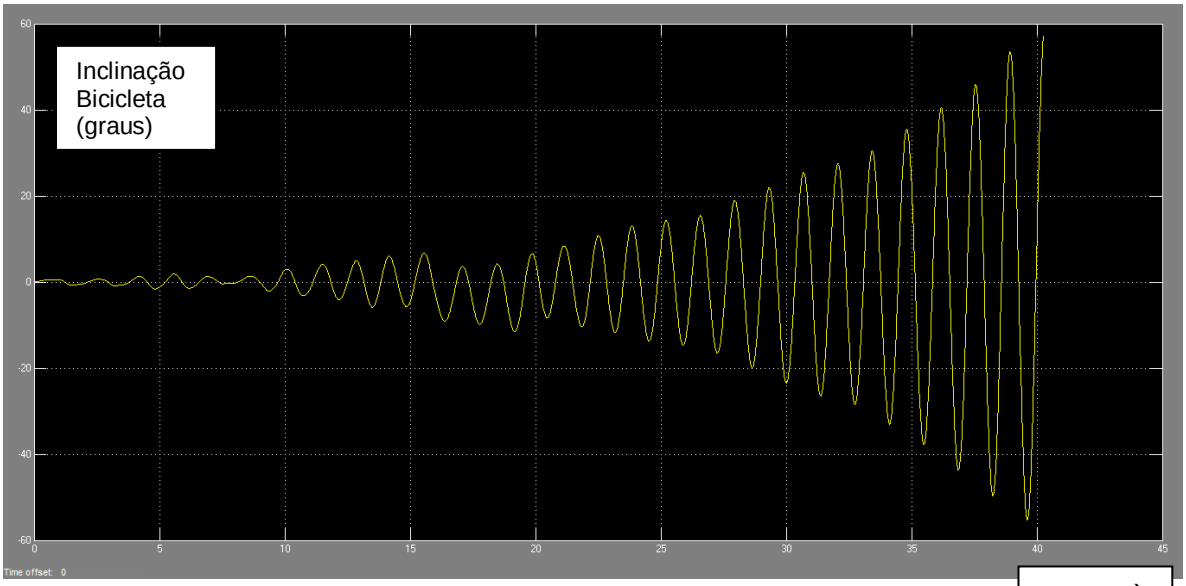


Figura 5.16 - Inclinação com Controlador P, Kp=20;

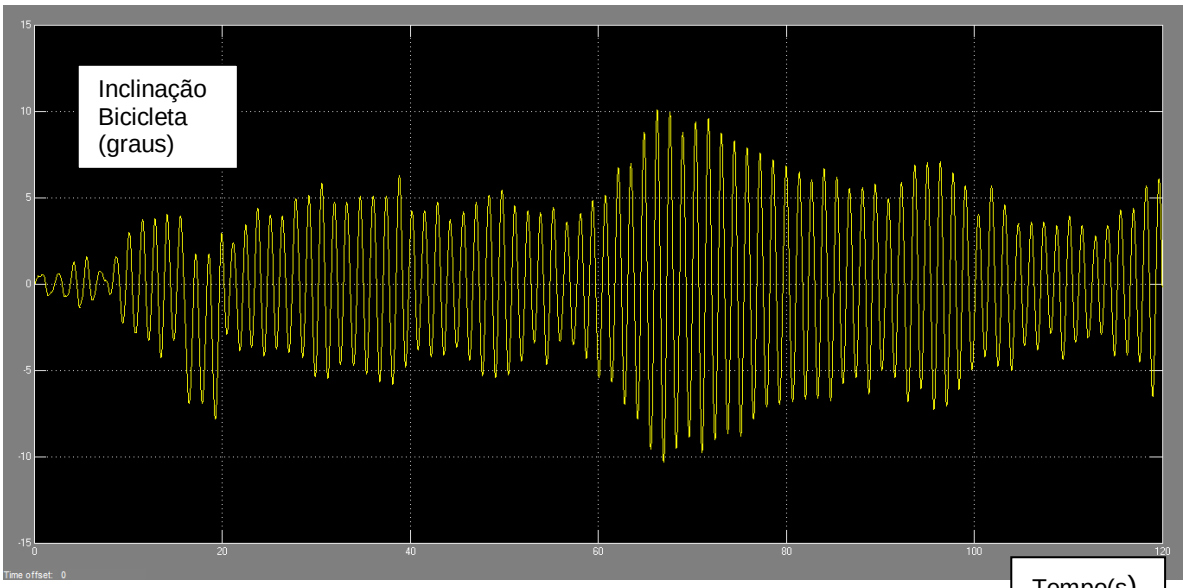


Figura 5.17 - Inclinação com Controlador P, Kp=22;

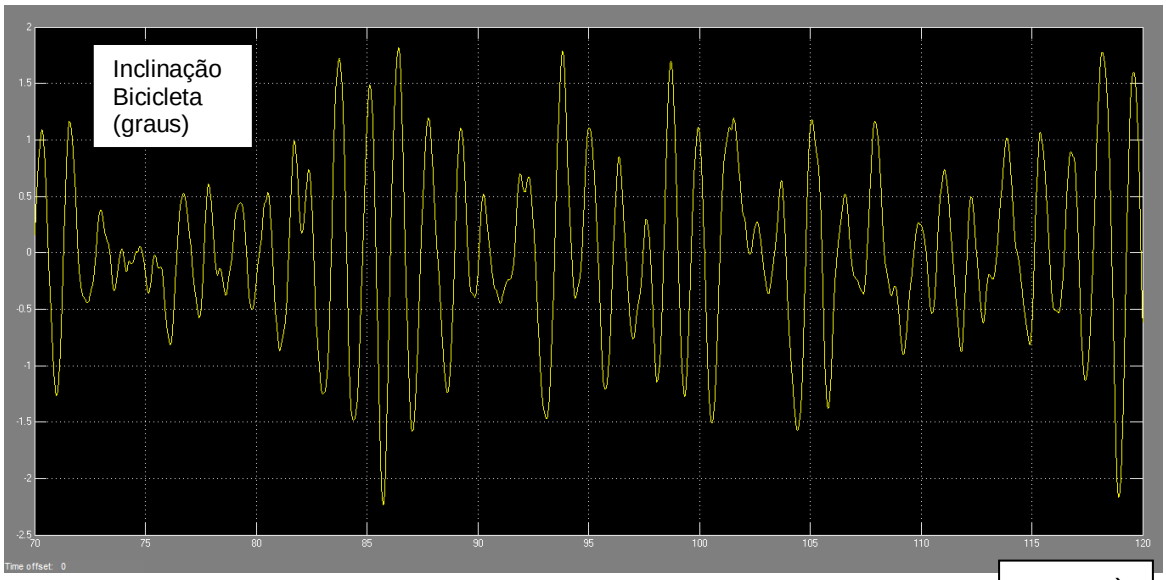


Figura 5.18 - Inclinação com Controlador P, Kp=30;

Após a apresentação destes resultados é possível verificar que um controlador proporcional é incapaz de controlar o sistema, quando este possuí um ganho de 20. Este sistema torna-se estável quando o ganho for superior a 22, podendo assim controlar o equilíbrio da bicicleta. Este valor é passível de ser praticável no sistema real apesar de possuir um ganho de magnitude um pouco elevada, gerando assim sinais de controlo de ordens não adequadas ao sistema, oscilando bastante. A solução encontrada utilizando um controlador PID, apresenta então uma melhor performance comparada com este.

5.2.4 Controlador PID com Controlador de Mapeamento

Num último nível, foi criado um ambiente de simulação que, para além do controlo de equilíbrio por algoritmo PID, efetue um controlo de posição e deslocamento efetuando apenas a atuação da direção. O controlador PID demonstrou nas simulações efetuadas, do ponto de vista de controlabilidade e robustez, ser o mais indicado para o controlo de equilíbrio, e será adaptado de igual modo para o controlo do mapeamento desejado. A *Figura 5.19* pretende representar esse ambiente de simulação, num contexto hipotético. Percebe-se perfeitamente que este controlo adicional de direção impõe mais um grau de liberdade ao sistema geral, o que condiciona o comportamento do controlador inicial.

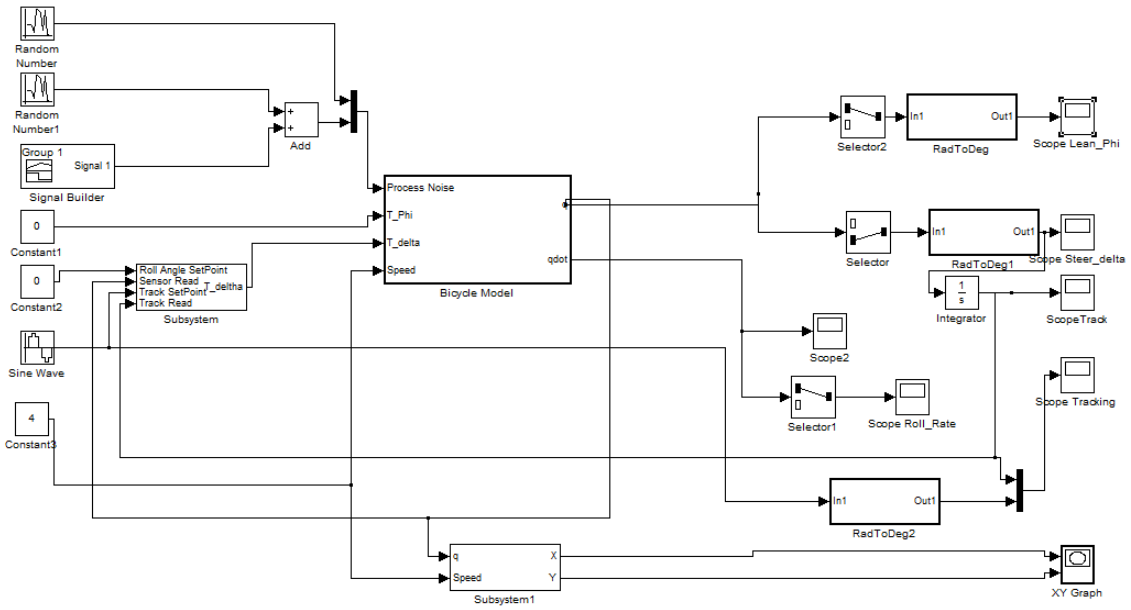


Figura 5.19 - Controlador PID + Tracking;

5.2.4.1 Bloco Controlo PID + Tracking

O ambiente de simulação da *Figura 5.5* foi modificado de forma a poder executar o controlo de mapeamento. O bloco de controlo geral irá então executar os dois algoritmos PID, para equilíbrio e deslocamento. Este recebe os valores lidos do sensor, nomeadamente a inclinação e o estado do volante, e após a execução deste algoritmo, devolve um valor de rotação da direção, em radianos. Este por sua vez compara a posição criada pelo movimento da bicicleta com uma função que simule o deslocamento pretendido, tratando-se nesta simulação de um seno com amplitude 0.5 e período 10 segundos. O bloco está representado na *Figura 5.20*. De referir que se trata apenas de uma solução de baixo custo, tentando beneficiar apenas do hardware instalado para cumprir com este objetivo. Será apresentada então, uma cascata de dois controladores, onde o primeiro possui valores diferentes para os parâmetros do controlador de equilíbrio,

isto devido à interação com o segundo controlador PID, que pretende representar o controlo de deslocamento do sistema. Considerou-se então os seguintes parâmetros, *Kp1 = 6, Ki1= 2, Kd1= 5, Kp2 = 3, Ki2 = 0.02, Kd2 = 20,* a uma velocidade de 4 m/s.

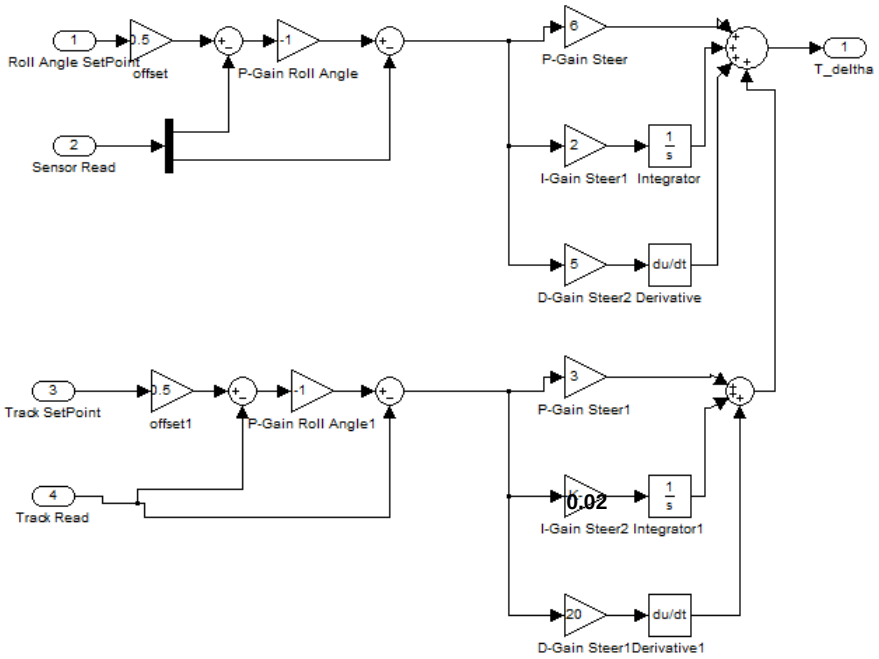


Figura 5.20 - Bloco Controlador Equilíbrio e Mapeamento;

5.2.4.2 Bloco de Mapeamento XY

De igual modo, foi criado um bloco que consiga mapear em X e Y a posição exata da bicicleta durante a simulação. Este, como seria de esperar, é semelhante ao obtido através da integração dos graus de variação do volante ao longo do tempo. O bloco encontra-se representado na *Figura 5.21*.

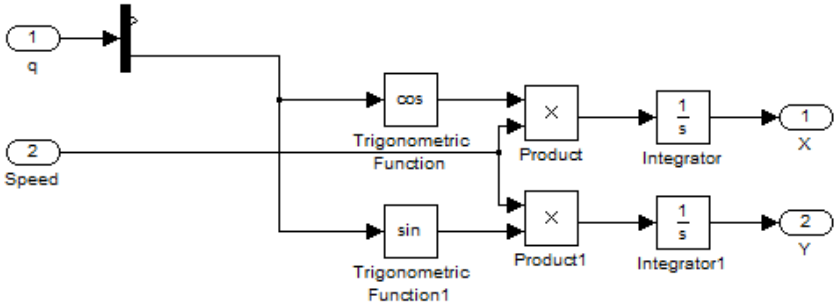


Figura 5.21 - Bloco de Mapeamento XY;

5.2.4.3 Simulações sem Perturbações externas

Assumindo um cenário de teste onde o *Signal Builder* esteja desligado, ou seja, não existe uma perturbação externa aplicada ao sistema, apenas o ruído criado por atrito, efetua-se uma simulação durante um período de 120 segundos. A *Figura 5.22* representa o equilíbrio controlado da bicicleta, enquanto que a *Figura 5.23* representa o controlo da posição.

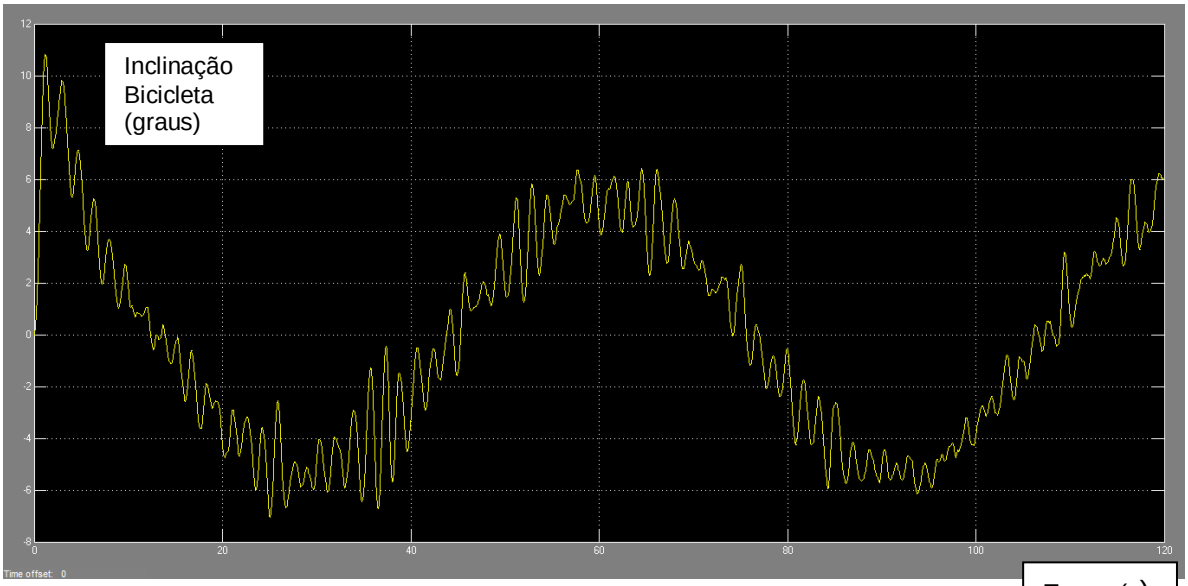


Figura 5.22 - Equilíbrio sem perturbação, Controlo PID + Mapeamento ;

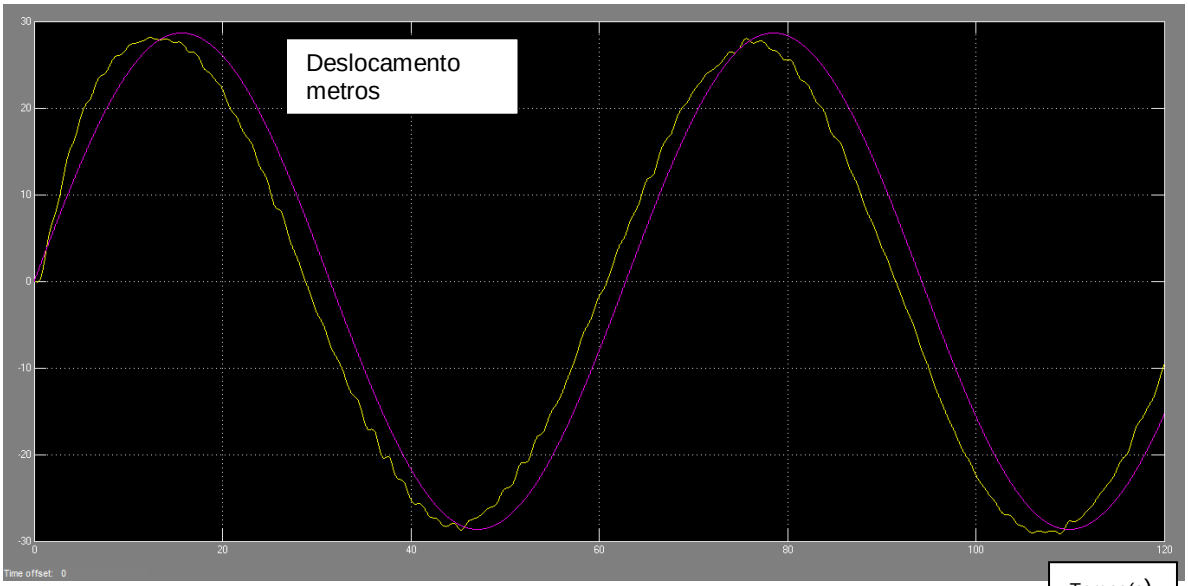


Figura 5.23 - Mapeamento sem perturbação, Controlo PID + Mapeamento;

Através da simples análise da *Figura 5.22* supra indicada, é possível verificar a continuidade do controlo de equilíbrio, assim como o de mapeamento. Na *Figura 5.23* é possível ver a roxo o sinal com o deslocamento pretendido e a amarelo o deslocamento real controlado pelo *bloco 5.2.4,* sendo este resultado muito semelhante, apresentando apenas um ligeiro atraso em relação ao deslocamento pretendido.

5.2.4.4 Simulações com Perturbação externa

Assumindo um cenário de teste onde o *Signal Builder* esteja ligado, ou seja, criando uma perturbação externa aplicada ao sistema de magnitude -0.5 radianos, efetuou-se uma simulação durante um período de 120 segundos. A *Figura 5.24* representa o equilíbrio controlado da bicicleta, enquanto que a *Figura 5.25* representa o controlo da posição.

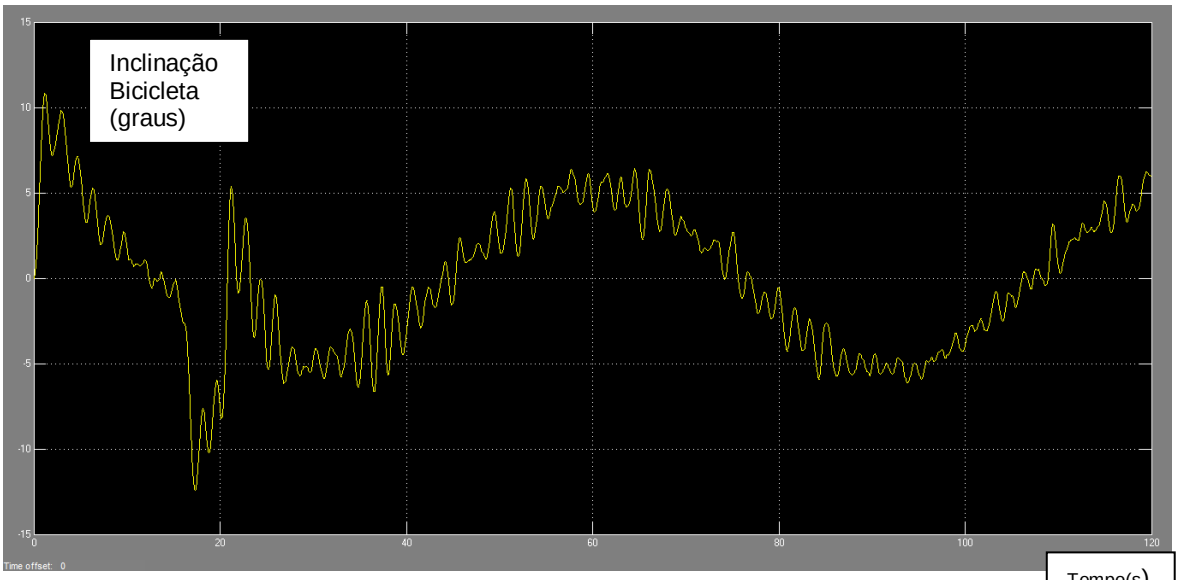


Figura 5.24 - Inclinação com perturbação -0.5 rad, Controlo PID + Mapeamento;

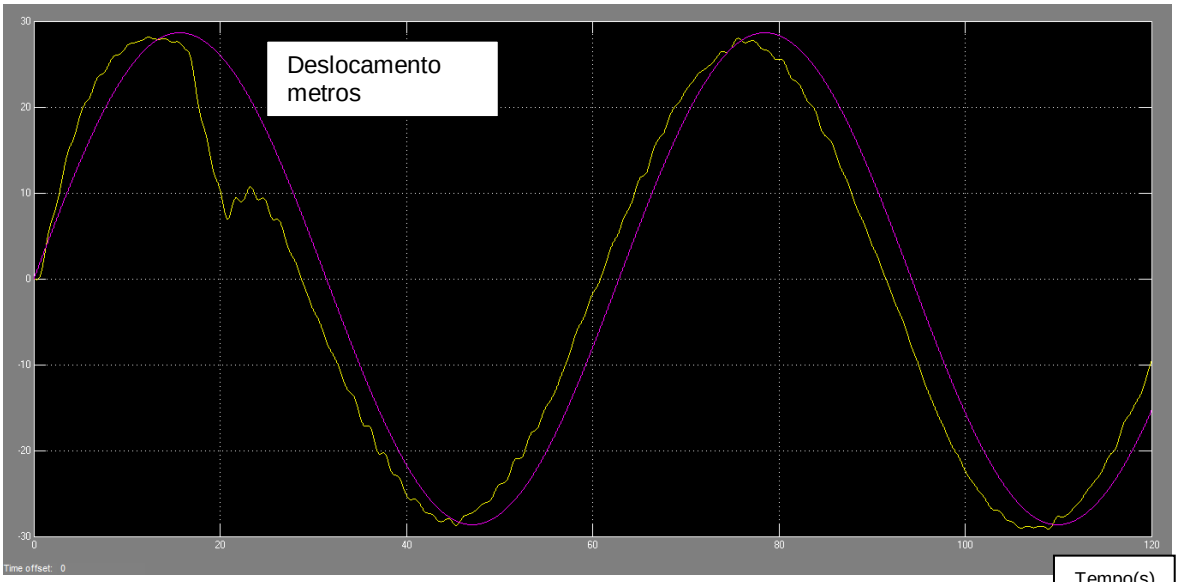


Figura 5.25 - Mapeamento com perturbação -0.5 rad, Controlo PID + Mapeamento;

A análise é semelhante à efetuada no *ponto 5.2.4.3*, onde se pode visar que o controlador consegue recuperar a posição desejada, mesmo com perturbação externa aplicada ao sistema. Esta conclusão abre portas a uma implementação deste controlador no mundo real. Apesar do ponto fundamental desta tese incidir apenas no controlo de equilíbrio da bicicleta, a análise deste ambiente de simulação é academicamente interessante e incentiva, num futuro próximo, a instalação deste sistema no controlo atual.

5.2.5 Resultados para diferentes Frequências

Como os parâmetros de controlo dependem diretamente do passo de amostragem escolhido, é curioso efetuar um pequeno estudo e possível obtenção dos mesmos para uma gama de frequências que possam ser usadas no modelo real de teste. A *tabela 5.1* representa conjuntos de parâmetros capazes de controlar o sistema de equilíbrio, para os diferentes passos de amostragem a uma velocidade fixa de 4 m/s.

f (Hz)	*h (s)*	Kp	Ki	Kd
10	0.1	5	0.001	8
100	0.01	4.2	0.003	11
500	0.002	4.2	0.002	4

Tabela 5.1 - Parâmetros vs. Passo de amostragem;

Estes valores servem de guia para os ajustes nos cenários de teste reais, onde a bicicleta irá efetuar uma viagem controlada.

5.2.6 Discussão e Comparação dos controladores simulados

Como pode ser visto, a partir dos resultados das simulações, tanto o controlador proporcional, P, como o controlador PID são capazes de equilibrar a bicicleta em ambiente de simulação. Ambos obtêm respostas aceitáveis dadas as perturbações externas, conseguindo controlar o sistema de forma adequada, apesar do controlador proporcional necessitar de um ganho mais elevado. Assim a solução implementada com o controlador PID é a mais indicada à realização física, possibilitando assim, um algoritmo rápido, eficiente e robusto, não sendo dependente do modelo matemático que representa a bicicleta. Com o decorrer das simulações e testes físicos do sistema, deparou-se que o facto de reduzir a componente integral do controlador melhorava, em certo ponto, a resposta do sistema. O controlador assemelha-se então a um controlador do tipo PD, não puro, por possuir uma componente integral infinitesimal, sendo este típico de sistemas mais dinâmicos. A componente derivativa quando combinada com a ação proporcional tem justamente a função de prever a ação de controlo para o sistema reagir mais rapidamente, necessidade pretendida ao modelo aplicado. Sabe-se também que a ação derivativa fornece uma correção antecipada do erro, diminuindo o tempo de

resposta e melhorando a estabilidade do sistema atuando em intervalos de tempo regulares. Uma vez que este tempo é inversamente proporcional à velocidade de variação da variável controlada leva a que num caso com forte componente derivativa o processo fica extremamente sensível ao ruído podendo levar a instabilidade. Relativamente à obtenção concreta destes parâmetros de controlo, esta foi baseada num bloco de sintonização automática PID presente no ambiente de simulação, *Simulink,* que é capaz de gerar parâmetros que permitam controlar o sistema. A afinação até se obter os finais foi feita através de tentativa e erro, tendo por base as noções de controlabilidade.

Capítulo 6

Bicicleta Real

Este capítulo visa descrever o trabalho realizado para implementar o controlador de equilíbrio na bicicleta física. Uma vez que os parâmetros dependem do valor da velocidade, esta precisa de ser controlada. Com esse mesmo intuito irá ser efetuada a análise da implementação do respetivo controlador de velocidade, que permite à bicicleta movimentar-se com um valor constante, este definido *à priori*. O controlador PID, simulado no *Capítulo 5,* foi capaz de equilibrar o modelo representativo da bicicleta nesse ambiente de simulação. Neste capítulo é apresentado a implementação do sistema na bicicleta real. Para além da descrição do funcionamento do sistema global, irá ser fornecida a informação mais pormenorizada sobre o funcionamento do software utilizado para o efeito. Por fim, serão explicitados os resultados finais, apresentando o funcionamento do sistema no mundo real.

6.1 Descrição Geral de Funcionamento

O modelo físico do sistema pode ser divido, como já foi explicitado, em dois subsistemas, um para controlo de equilíbrio e outro para o controlo de velocidade. O funcionamento deste último pode ser assumido como independente do primeiro. Como é óbvio, no caso extremo da bicicleta perder o equilíbrio e cair, este controlador não poderá executar a sua função de manter a velocidade da bicicleta. Em relação ao controlador de equilíbrio, este para funcionar depende da velocidade fixada *à priori*, pois os parâmetros não são dinâmicos, tornando-se assim totalmente dependente do controlador de velocidade. A *Figura 6.1* representa o sistema do equilíbrio aplicado.

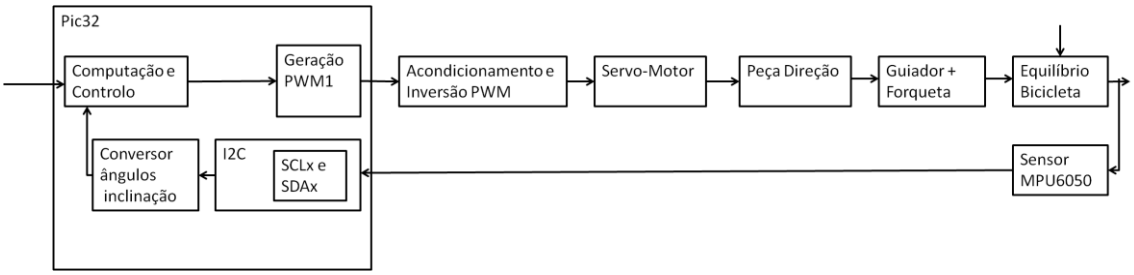


Figura 6.1 - Sistema de Equilíbrio;

Após ocorrer uma perturbação externa aplicada ao sistema, por declive ou atrito da estrada, força do vento, entre outros, o algoritmo que efetua a leitura do sensor a uma frequência programada, deteta uma inclinação no sistema com um determinado valor, obtido em graus. Para esta obtenção é necessário executar uma função capaz de converter os valores enviados pelo sensor, *RAW*, em ângulos. Esta função trata-se do código descrito no *Capítulo 3.2.2.1a*, e foi verificada utilizando uma ferramenta construída para esse propósito, *Figura 6.2.* Trata-se de um utensilio simples, onde após estar nivelada, é capaz de colocar o sistema num determinado ângulo, medido com um transferidor. Este é comparado com o valor lido, comprovando assim a veracidade dos valores obtidos.

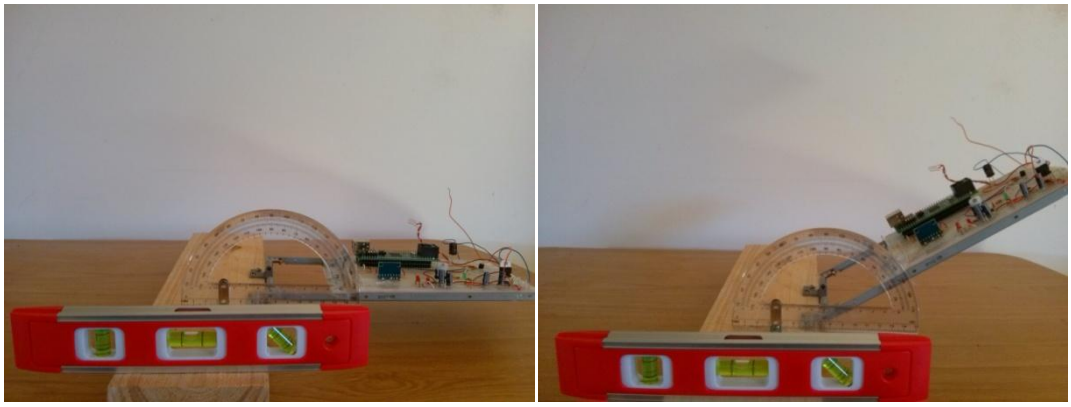


Figura 6.2 - Conversão dos valores RAW lidos do sensor em angulos;

Estes valores *RAW,* que são recebidos por comunicação I2C, previamente estabelecida, são então convertidos em ângulos e analisados pelo algoritmo de controlo. O controlador PID, com parâmetros fixos para uma dada velocidade, irá comparar o valor lido pelos sensores com o *Setpoint*, gerando como resultado um sinal de controlo adequado. Este sinal de controlo, dado em ângulos de rotação, é por sua vez convertido num sinal PWM invertido que é enviado para o servo, passando primeiro pelo bloco de acondicionamento e inversão do sinal. Assim, ocorre uma atuação no volante e direção, controlando o equilíbrio do sistema, repetindo-se este ciclo durante o tempo de teste desejado. Após este tempo ser esgotado, o sistema desativa o motor de tração, cujo funcionamento e controlo está descrito na *Figura 6.3*. O detetor da velocidade instantânea está a cargo do sensor de infravermelhos, como já descrito em 3.1.5.1. O seu funcionamento, como vimos, passa por um simples contador de pulsos que integrado num determinado tempo calcula o número de rotações por minuto, RPM, valor facilmente convertido em metros por segundo, tendo o conhecimento do raio da roda dentada.

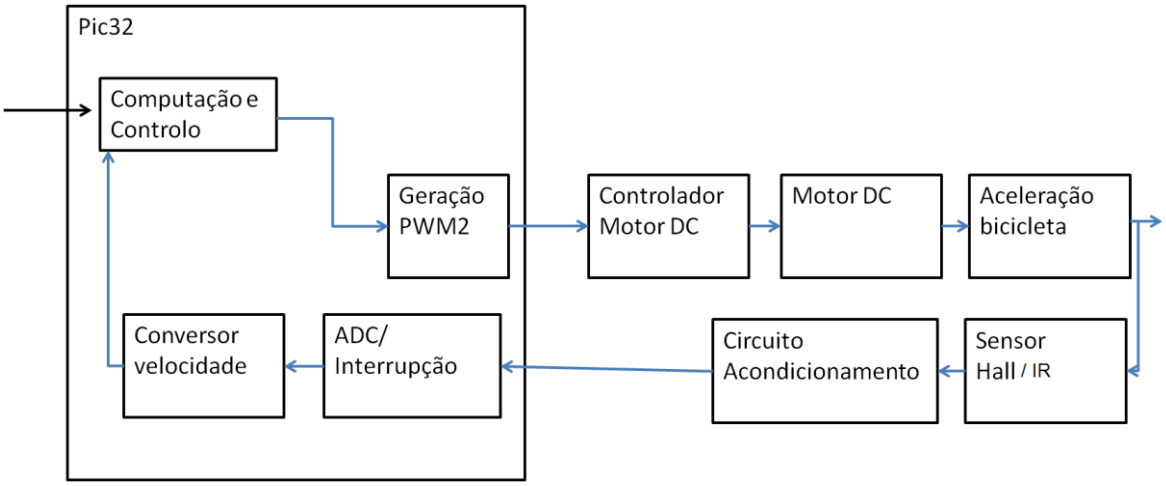


Figura 6.3 - Sistema de Velocidade;

A velocidade lida é comparada com o *Setpoint* da velocidade pretendida, através de um algoritmo de controlo PI, capaz de gerar o sinal PWM adequado ao controlador do motor DC. Este último é o responsável por adequar as tensões do motor à velocidade requisitada. Assim que o tempo de teste acabar, pode ser enviado um sinal que força o motor DC a desligar, desacelerando a bicicleta com o atrito criado.

6.2 Funcionamento do Motor DC

O funcionamento do motor DC é da responsabilidade do controlador específico aplicado ao sistema. Não é do cerne desta tese, a explicação em detalhe do funcionamento desse, podendo apenas ser descrito o modo como ele atua por forma a acelerar a bicicleta. O circuito geral encontra-se apresentado em *Circuito 3.2a*. Este controlador funciona com base em sinais PWM recebidos, convertendo-os em tensão (o motor serve de integrador dos pulsos PWM) a aplicar ao motor e por consequência acelerar o sistema. Será então necessário criar um algoritmo capaz de converter a velocidade instantânea num sinal PWM adequado ao controlador do motor DC.

6.2.1 Controlador PI

O controlador implementado, de forma a criar um sinal PWM capaz de controlar a velocidade do motor, é um controlador PI, *Proportional Integrative*. Implementado com um conjunto de parâmetros adequados para o correto funcionamento do sistema. Este controlador permite à bicicleta partir do repouso e acelerar até atingir a velocidade pretendida, mantendo-a assim que esta for atingida. Em seguida é apresentado o *pseudo-código 6.1* criado para tal função.

```
// Durante o Funcionamento do Sistema

// Se a Flag de controlo for ativada, ou seja ocorreu uma interrupção gerada pelo timer especifico que está a cargo do ciclo de controlo.

v = getV();          // Lê a velocidade instantânea dada pelo sensor de IR;

//Se a velocidade atual for superior à velocidade pré-definida à priori, desativa o sistema de equilíbrio auxiliar, caso este exista;

if(v>veloc-1.0)
   TurnOFF();
else
   TurnON();

PI(v,veloc);          // Algoritmo, que irá comparar v, com a velocidade, veloc, pretendida;

// Reset da Flag;
```

Pseudo-Código 6.1 - Funcionamento do Sistema, Velocidade;

O controlador PI aplicado ao sistema é criado então pelo *pseudo-código* 6.2 apresentado em seguida.

```
void PI(float velAtual,float velPoint)
{

// Inicialização e Variáveis de Controlo;
        u = 0;  //Sinal Controlo
        p = 0;  //Proporcional
        i = 0;   //Integral

// Erro atual:
        erro_actual = velPoint - velAtual;

// Termo proporcional:
        p = KpDC*erro_actual;

// Termo integral:
        i = i_anteriorDC + erro_actual*KiDC;

// Atualizações:
        erro_anteriorDC = erro_actual;
        i_anteriorDC = i;
```

```
        u = p+i;

        velocControl(u);     // Envia o sinal de controlo PWM para o controlador externo;
}
```

Pseudo-Código 6.2 Controlador PI, Velocidade;

Foi necessário incluir uma função capaz de ler a velocidade instantânea, obtida através dos valores recebidos do sensor de infravermelhos. O funcionamento desta função passa pela contagem dos pulsos criados pela passagem de cada dente da roda dentada pelo sensor. O pulso gera uma interrupção externa no microcontrolador e efetua a contagem. Após a contagem de um determinado número de pulsos, equivalente ao número de dentes da roda e a diferença de tempo, obtém-se um valor para as rotações por minuto, RPM. Após obter este valor, a conversão em metros por segundo é elementar. O *pseudo-código 6.3 e 6.4* pretende demonstrar como tal função foi configurada e criada.

```
void setup_Tach()            // Configura o módulo de interrupções externas do sistema
{
  INTCONbits.INT1EP = 1; //Setup INT1 para criar a interrupção na transição negativa;
  IFS0bits.INT1IF = 0;       // Reset INT1 interrupt flag;
  IEC0bits.INT1IE = 1;       // Ativa INT1 a ISR onde a flag é posta a 1;
  IPC1bits.INT1IP = 4;       // Prioridade;
}
```

Pseudo-Código 6.3 Configuração Interrupções Externas;

```
// Durante o Funcionamento do Sistema;
// Rotina de Interrupção Ativa;

// Inicializações:
       double delta = 0;
       double RPM = 0;
       double raio = 0.077;

// Primeiro Pulso:

       if(ContPulse == 0)
             {
                WriteCoreTimer(0);
                Tinicio = ReadCoreTimer();
             }

         ContPulse++;
```

```
// Quando contar 36 Pulsos == 36 dentes, o que equivale a uma rotação.

        if(ContPulse ==36)
        {
          delta = ((ReadCoreTimer() - Tinicio))/SYSCLK;
          RPM = ((60)/(delta));
          Veloc = raio * RPM * 0.10472;
          ContPulse = 0;
          WriteCoreTimer(0);
        }

        flagTACH=0;                          // Reinicia a flag de interrupção;
```

Pseudo-Código 6.4 Função da Velocidade Instantânea;

6.2.2 Velocidade Máxima

O motor DC encontra-se alimentado à tensão de 12V, como já descrito, sendo este capaz de acelerar a bicicleta à sua velocidade máxima num intervalo de tempo de 8 a 10 segundos. Estes testes foram realizados em estrada, sendo que a velocidade máxima atingida, numa média de ensaios, tem um valor de 3.86 m/s. É possível admitir que o sistema é capaz de atingir 4 m/s como velocidade de pico.

6.2.3 Resposta do Controlador PI de Velocidade

O sistema descrito que executa o controlo de velocidade da bicicleta foi testado num cenário construído com esse propósito. Encontra-se assim a bicicleta parada, com a roda de tração levantada, para que o sistema não se movimente. As funções de teste foram executadas para uma velocidade pré-definida com valor de 3.8 m/s, com o objetivo de assemelhar-se ao valor usado nos ensaios de equilíbrio. Foram lidos pontos através da ligação série do microcontrolador para a velocidade ao longo do tempo e estes foram interpolados de forma a obter a resposta apresentada na *Figura 6.4* (velocidade instantânea em função do tempo). É possível verificar o correto funcionamento do controlador, atingindo um *overshoot* aceitável e dentro da gama de velocidades controláveis. O mesmo acontece quando se aplica um "degrau" de atrito externo ao sistema aproximadamente aos 27 segundos. O controlador é capaz de compensar o atrito e levar a velocidade instantânea à pretendida, esta com um valor próximo dos 3.8 m/s.

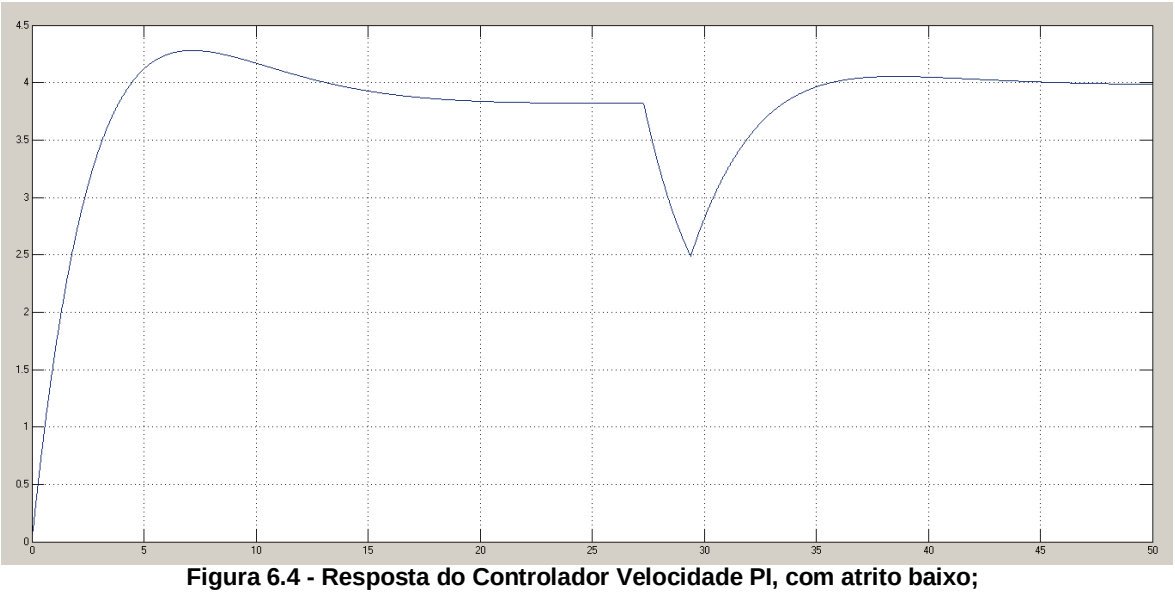


Figura 6.4 - Resposta do Controlador Velocidade PI, com atrito baixo;

No entanto, nos ensaios já realizados em estrada foi notório a falta de potência do motor, criada pelo peso excessivo da bicicleta em relação à bateria utilizada e ao elevado atrito. Assim de forma a representar os ensaios em estrada aplicou-se à roda traseira um atrito constante de maior amplitude. Na *Figura 6.5* encontra-se a resposta (velocidade instantânea em função do tempo) que mais se assemelha ao movimento em estrada, onde o controlador possui um *settling time* mais elevado (tempo de aceleração até 3.8 m/s de cerca 8 segundos), não existindo *overshoot,* e tendo dificuldades em atingir velocidades superiores a 3.8 m/s. Após a aplicação de atrito externo este consegue recuperar, embora a velocidade baixe para valores próximos do limite de controlabilidade de equilíbrio, 3.1 m/s.

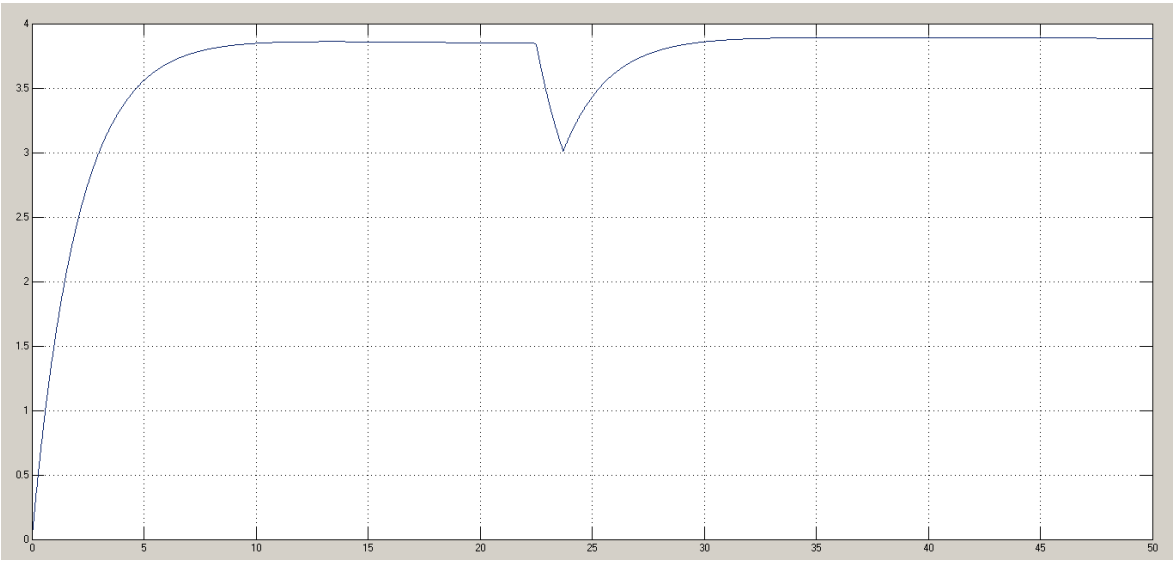


Figura 6.5 - Resposta do Controlador Velocidade PI em Estrada;

6.3 Bicicleta Equilibrada

Para o funcionamento do sistema de equilíbrio da bicicleta, foi implementado um controlador PID cujo foi simulado no *capítulo 5*. No modelo de teste, a posição da direção é controlada pelo aplicar de um binário externo ao eixo de direção, enquanto que na bicicleta física, o binário é aplicado primeiramente a uma peça de direção, atuando esta de seguida no eixo. Por não nos encontrar-mos num ambiente de teste ideal, linear, e idêntico a um ambiente de simulação em *Simulink*, os parâmetros de controlo do sistema físico, tiveram de ser ajustados para o funcionamento do módulo de equilíbrio, uma vez que como já foi referido, os ângulos criados no servo não correspondem aos ângulos obtidos na direção. O *pseudo-código 6.4* pretende demonstrar o funcionamento do algoritmo de controlo PID, utilizado para o controlo de equilíbrio.

```
void PID(float atual,float setPoint)
{

// Declaração das variáveis de controlo;

        u = 0;  //Sinal Controlo
        p = 0;  //Proporcional
        i = 0;   //Integral
        d= 0;   //Derivativo

// Erro atual:
        erro_actual = setPoint - atual;
// Termo proporcional
        p = Kp*erro_actual;

// Termo Integral:
        i = i_anterior + erro_actual*Ki;

// Termo Derivativo:
        d = Kd*(erro_actual-erro_anterior);

// Atualizações:
        erro_anterior = erro_actual;
        i_anterior = i;

// Sinal Controlo
        u = p+i+d;
        viraMotor(u);  // Envia o sinal de controlo PWM para o Servo;
}
```

Pseudo-Código 6.4 Controlador PID, Equilíbrio;

O funcionamento geral deste controlador está a cargo da função representada pelo *pseudo-código 6.5,* cujo objetivo é criar o ambiente de testes, inicializando os módulos necessários e criando o algoritmo adicional que o controlador PID de equilíbrio requisita.

```
void Equilibrium (void)
{

// Configurações dos módulos necessários:

   Setup_I2C();                 //Comunicação I2C
   Setup_MPU6050();         //Registos e Funções do Sensor
   Setup_PWM();               //Timer e Output Compare para o Servo
   Setup_Timer4();           //Timer de Controlo
   Zero_Sensors();          //Inicialização de variáveis do Giroscópio e acelerómetro;
   Setup_DCmotor();         //Inicialização do Timer responsável pelo PWM do motor DC;
   INTEnableSystemMultiVectoredInt(); //Ativa Interrupções

// Declaração Condições Iniciais;

// Levar a Bicicleta à velocidade adequada de funcionamento dos parâmetros;

// Durante o Funcionamento do Sistema

// Se a Flag do controlador de velocidade for ativada, ou seja ocorreu uma interrupção
gerada pelo timer especifico para este controlador.


if (flag == 1)
      {
      Get_Accel_Values();     //Recebe valores do Acelerómetro;
      Get_Accel_Angles();     //Converte esse valores em ângulos;
      act = ACCEL_YANGLE;
      PID(act,offs);               //Angulo offset indicado para o cenário de teste atual;

      velocControl(veloc);      //Manter a velocidade controlada;
      }

// Reset Flag;
}
```

Pseudo-Código 6.5 Funcionamento do Sistema, Equilibrio;

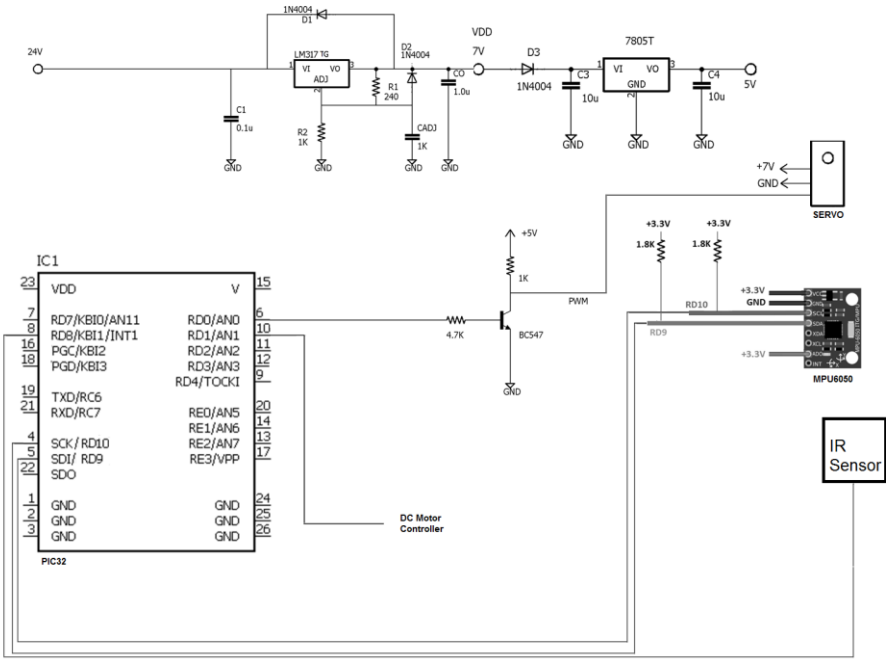
6.3.1 Travagem

A roda traseira da bicicleta possui um eixo de rotação livre, portanto a possibilidade de retardar fisicamente o sistema geral através deste, torna-se praticamente impossível. Existe então a necessidade de equipar um sistema de travagem à bicicleta. No cenário atual, a travagem é feita passivamente, onde após a finalização do tempo de teste, ou simplesmente pela desativação manual do controlador, o motor DC é desligado, bem como o sistema de equilíbrio. A bicicleta acaba por desacelerar pelo atrito criado, podendo ser agarrada de forma a não danificar o equipamento.

6.3.2 Trem de Aterragem

O trem de aterragem trata-se de um sistema cuja implementação, foi ponderada inicialmente, onde se criou uma simples função de ativação. Este seria automaticamente ativado quando a velocidade instantânea atingisse um limite inferior, impossibilitando o correto funcionamento do controlador de equilíbrio, para os parâmetros fixos utilizados no sistema. Seria usado também para levar a bicicleta à velocidade pretendida e para a desaceleração em equilíbrio do sistema. O sistema passaria simplesmente pela ativação/desativação de um porto do microcontrolador, funcionando de interruptor para este.

6.3.3 Esquema Elétrico Geral



Circuito 6.1 - Circuito Elétrico Geral;

6.4 Resultados

Nesta secção serão finalmente apresentados os resultados finais do sistema físico. De forma a respeitar algumas das condições utilizadas nas simulações, e por outro lado, criar desafios de maior dificuldade ao sistema, foram efetuados testes em dois cenários diferentes, ambos partindo de resultados e noções retiradas do que já foi estudado anteriormente. Assim, optou-se por efetuar testes num pavilhão gimnodesportivo e testes em estrada.

6.4.1 Teste em Pavilhão Gimnodesportivo

De forma a representar o cenário ideal de funcionamento optou-se por efetuar testes num pavilhão gimnodesportivo, de onde se poderia concluir imediatamente a funcionalidade, ou incapacidade do sistema, caso este aqui não funcionasse corretamente. Torna-se assim numa condição necessária e suficiente ao objetivo proposto, podendo ser retiradas noções importantes para testes com um grau de maior dificuldade.

Figura 6.6 - Ensaios em pavilhão Gimnodesportivo;

6.4.1.1 Características do Cenário de Teste

Por se tratar de um pavilhão gimnodesportivo retratou-se na perfeição o cenário simulado em *MATLAB* aquando não são aplicadas forças externas de grande magnitude, *Simulação 5.2.2.2.* É então recriado um espaço fechado, sem vento, piso liso totalmente plano, possibilitando tração suficiente sem deslizamento das rodas e com espaço suficiente para uma viagem prolongada. O espaço amplo é útil de forma a permitir a livre movimentação da bicicleta sem obstáculos.

6.4.1.2 Condições de Funcionamento

O funcionamento do sistema está dependente de diversos fatores. Como a bicicleta não possui sistema de aterragem, esta terá de ser segura até atingir a velocidade pretendida. Este aspeto requer um certo tempo, onde só após a velocidade ser atingida é ativado o controlo de equilíbrio. O sistema geral é inicializado e finalizado, ligando/cortando a energia fornecida aos sistemas de controlo e ao motor, pelas baterias. O ciclo de controlo foi de 100 Hz, de modo a corresponder ao passo de amostragem utilizado nas simulações e assim se obter um ponto de partida para os parâmetros do controlador. Devido ao peso excessivo da bicicleta e à potência capaz de ser aplicada ao motor, esta demoraria muito tempo a acelerar, facto contraposto por um simples empurrão inicial aplicado em todos os ensaios, não influenciando as condições de teste. Devido a essa mesma falta de potência a malha de controlo PI e respetivo tacómetro acabam por ser insignificantes para o funcionamento do sistema, já que a bicicleta não atinge velocidades numa gama muito superior à pretendida. Esta teve de circular a uma velocidade máxima controlada de aproximadamente 3.1 m/s, comparativamente ao pico de 3.8 m/s capaz de ser produzido, onde os parâmetros de controlo serão idênticos para ambas velocidades. Em alguns ensaios, optou-se inclusive por desligar estes módulos e aplicar ao motor a potência máxima desde o inicio do teste. Assim, iniciou-se os primeiros testes ao protótipo criado, para exemplificar e provar o conceito proposto e projetado nesta tese. A função *main* do código de programação utilizado encontra-se descrita no *Apêndice A.2*.

6.4.1.3 Afinamento de Parâmetros

Como a frequência escolhida para o sistema corresponde ao passo de amostragem simulado, optou-se obviamente por testar inicialmente o sistema utilizando os parâmetros de controlo correspondentes à velocidade escolhida. No entanto, de forma a este funcionar corretamente foi necessário uma pequena afinação dos mesmos, uma vez que os ângulos da direção não correspondem exatamente aos ângulos aplicados pelo servo, como se efetua nas simulações. A *Tabela 6.1* pretende explicar o raciocínio lógico de afinação, após uma análise ao sistema com os parâmetros do teste anterior.

Kp	Ki	Kd	Resultado	Comentário
9	0.02	12	X	• Demasiado Kp e Ki; • Ângulos muito elevados; • Não se equilibra;
4.5	0.003	12	X	• Necessita Kp superior; • Ângulos curtos; • Não se equilibra;
5.5	0.003	12	+/-	• Alguns momentos de equilíbrio; • Necessita Kp superior;
6	0.001	12	V	• Equilibra-se durante um certo tempo; • Oscila demasiado, e acaba por cair;
6.3	0.001	12	V	• Equilibra-se totalmente; • Mesmo oscilando, recupera o equilíbrio; • Comportamento semelhante à simulação;

Tabela 6.1 - Afinação dos Parâmetros de Controlo;

6.4.1.4 Comportamento do Sistema

Para os parâmetros encontrados, Kp = 6.3, Ki = 0.001, Kd = 12, o sistema comporta-se como pretendido, cumprindo assim o objetivo principal desta tese, onde se colocou uma bicicleta a efetuar uma viagem autónoma em equilíbrio. O comportamento da mesma é semelhante ao visualizado nas simulações sem perturbações externas, *ponto 5.2.2.2*, não obstante o facto de os parâmetros apresentarem valores ligeiramente diferentes. A compensação do conjunto direção/servo pode explicar este desvio, bem como a falta de potência do motor DC, para em conjunto com a malha de controlo PI, manter a velocidade constante. Se a opção fosse o aumento da velocidade, nestas condições, o sistema de velocidade acabaria por funcionar em malha aberta, não conseguindo atingir uma velocidade superior à máxima de 3.8 m/s. Em funcionamento verifica-se as oscilações de equilíbrio esperadas, conseguindo a malha de controlo manter a bicicleta fora dos ângulos críticos, percorrendo uma distância considerável, em padrões distintos como seria de esperar. É de notar também a larga excursão do deslocamento lateral da bicicleta em andamento, considerando uma linha reta desde o ponto de partida. Após esta verificação resta testar o sistema em condições mais extremas, nomeadamente, numa estrada, tomando em consideração estes testes efetuados.

6.4.2 Teste em Estrada

Após estar cumprido o objetivo fundamental num cenário de testes ideal, é pertinente colocar o sistema sobre um conjunto de condições mais rigorosas, para que seja possível recriar os conjuntos de simulações *5.2.2.3 e 5.2.2.4.* A solução escolhida passou simplesmente por colocar o sistema em funcionamento numa estrada vulgar.

6.4.2.1 Características do Cenário de Teste

Este cenário pode ser resumido então a uma estrada, de piso alcatroado, sendo escolhido um troço em que não existam lombas, ou depressões, uma vez que a bicicleta não se encontra equipada com suspensão, podendo nesse caso danificar a estrutura e componentes. De duas faixas, esta possuí uma largura de aproximadamente 5 metros, descaindo do seu ponto central mais elevado, para as bermas numa amplitude de cerca 12º. Com alguma inclinação ao longo do percurso, permite o aumento da aceleração e velocidade exequível da bicicleta, já que serão efetuados os testes no troço descendente.

6.4.2.2 Condições de Funcionamento

Dos ensaios efetuados num cenário de teste ideal foram retiradas algumas informações sobre o funcionamento do sistema no mundo real. Destas, as que apresentam um maior grau de importância são as suas debilidades, que em condições mais extremas necessitam de ser analisadas e colmatadas. O sistema apesar de ter cumprido os requisitos, funcionou sobre condições restritas, provando a grande instabilidade nos ângulos de inclinação do modelo simulado. O ponto fulcral encontra-se no peso excessivo da bicicleta, que em conjugação com a pouca capacidade em fornecer potência ao motor, produz um sistema com uma baixa aceleração e velocidade máxima reduzida, apesar de esta ser suficiente para o controlo. Este aspeto poderia ser ultrapassado com a introdução de uma bateria adicional, o que devido ao tamanho reduzido da bicicleta e peso atual poderia ser contraproducente. Para combater este problema nestes ensaios, o tacómetro e o controlador PI de velocidade irão ser "desligados" (funcionando em malha aberta), impondo sempre a tensão máxima ao motor, uma vez que os parâmetros de controlo são admissíveis para a gama máxima de velocidade que a bicicleta poderá produzir. Os testes serão iniciados utilizando os parâmetros de controlo finais encontrados no cenário anterior, tendo a noção da possível necessidade de ajuste para tentar reduzir a excursão de deslocamento lateral, para que a bicicleta não embatesse nos passeios. Esta é então colocada no topo do troço, ao centro, e procede-se da forma já descrita no ponto *6.4.1.2*.

6.4.2.3 Queda e Substituição do Servomotor

Com os parâmetros de controlo obtidos do ensaio realizado em pavilhão efetuaram-se os primeiros testes no novo cenário descrito, resultando um destes, num forte embate da roda da frente no passeio. Nesse ensaio, com um início de movimento correto, a bicicleta acaba por sofrer o efeito do declive da estrada, que associado ao efeito do controlo, a direcionou para a berma. Não foi possível travar o movimento desta, acabando por embater no passeio. Resultou assim uma quebra irreparável no servomotor devido ao contragolpe sofrido da direção. Após este acidente optou-se por verificar se este poderia ter sido previsto no contexto de simulação. Recorde-se que em todas as simulações efetuadas para o sistema de controlo, considerou-se que a bicicleta circularia sobre um plano sem inclinação, apenas com perturbações. Assim, *à posteriori* simulou-se o sistema de controlo, utilizando os parâmetros idênticos ao ensaio que resultou no embate, Kp = 5.2, Ki = 0.003, Kd = 10, e assumindo agora a inclinação da estrada descrita (T_Phi). De forma a também representar os elementos externos adicionais, como o vento lateral, assumiu-se uma inclinação de magnitude 0.5 radianos, e uma velocidade mínima de 3.1 m/s. A *Figura 6.7* representa o equilíbrio da mesma ao longo do tempo, podendo ver que este o mantem, mesmo com a inclinação natural da estrada.

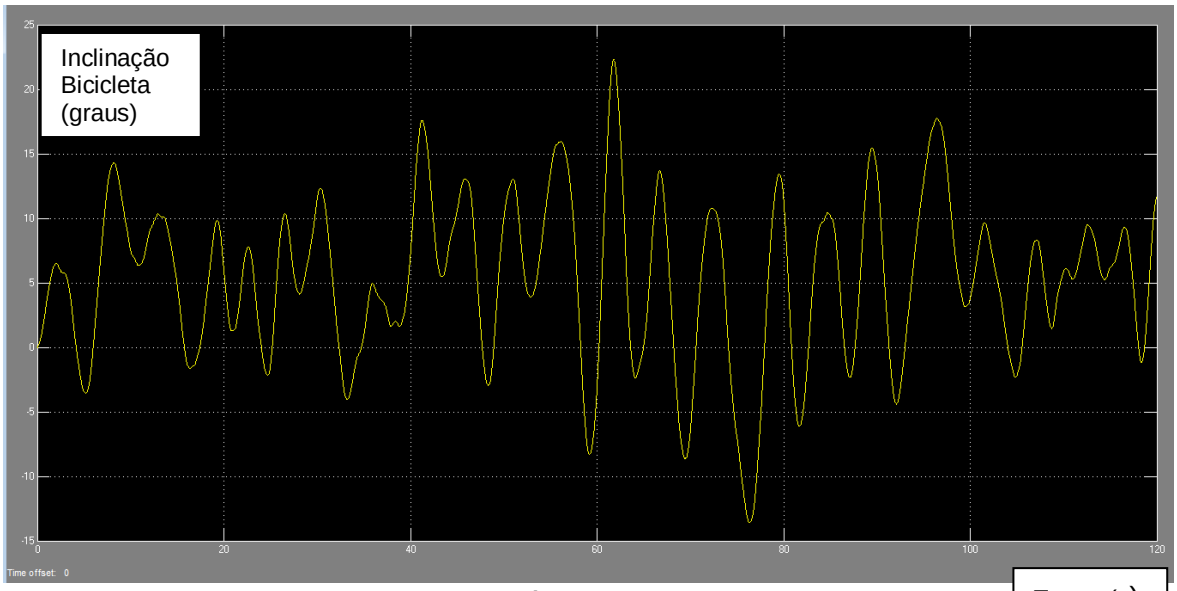


Figura 6.7 - Equilíbrio em estrada inclinada;

O problema encontrado não se trata no controlo do equilíbrio, mas sim na direção que tomou rumo à berma da estrada. Ao ser analisado o sistema, compreende-se que dada a inclinação, o sistema de controlo irá compensar movendo a direção da bicicleta consoante essa inclinação. A *Figura 6.8* representa a simulação efetuada comparando o ponto de embate simulado, e o obtido no ensaio real. Esta não considera o tempo e espaço necessário para a aceleração da bicicleta no ensaio real, e tendo sido obtida recorrendo ao bloco de mapeamento XY construído no *ponto 5.2.4.2.*

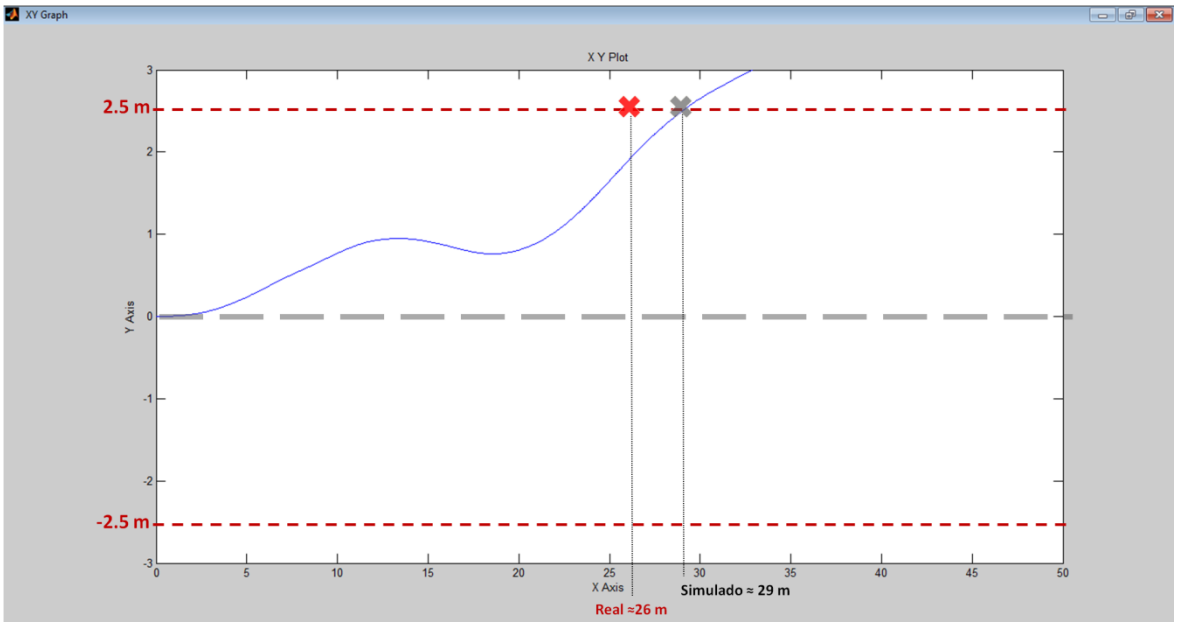


Figura 6.8 - Deslocamento Real vs. Simulado;

Como pode ser analisado na *Figura 6.8*, este episódio poderia ter sido previsto, já que no ambiente de simulações o embate com a berma acontece de igual modo. O ponto de embate real não é muito distante do simulado, comprovando assim, uma vez mais, o conceito construído nesta dissertação. Na *Figura 6.8* encontra-se assinalado a vermelho o ponto de embate real aproximado, podendo este ter uma pequena margem de erro. No ensaio real existiu também um pequeno *offset* positivo no ponto de partida da bicicleta, não partindo esta exatamente do zero no eixo dos *Y's.* Com o estado avançado do projeto, conclusões principais retiradas e devido ao preço elevado de um servomotor idêntico, optou-se pela reutilização de um já adquirido em projetos realizados ao longo do percurso académico, como substituto do inicial. Trata-se de um *Futaba s3003* [24], funcionando com pulsos *PWM* idênticos, não necessitando assim de reescrever código de programação. Com rolamentos em plástico não apresenta a robustez pretendida inicialmente, mas como se trata de um protótipo, este acaba por servir para os testes finais. Um aspeto negativo desta substituição, tirando a robustez, é a perca de torque na direção, já que este servo possui um binário de apenas 4.1kg/cm. Apesar desta característica, com a bicicleta em andamento, o binário debitado pelo *Futaba* [24] é suficiente para mover, em perfeitas condições, a direção. Antes das adaptações ao sistema inicialmente realizado, por via da avaria do servomotor, foram efetuados testes no cenário com as condições de funcionamento descritas. Em ensaios ainda realizados na presença de rodas de apoio, de forma a evitar inclinações extremas, foram testados conjuntos de parâmetros de controlo que permitissem a adaptação da bicicleta a uma estrada regular possibilitando assim uma viagem em equilíbrio prolongada no tempo. Este aspeto foi fundamental já que nos testes assistidos, utilizando um último conjunto de parâmetros fixo, Kp = 5.2, Ki = 0.003, Kd = 10, o protótipo mantinha o equilíbrio, sendo assim possível retirar algumas imagens, *Figura 6.9*.

Figura 6.9 - Viagem autónoma em estrada;

6.4.2.4 Comportamento do Sistema

Como referido na secção anterior, após um ensaio mais prolongado sem a intervenção direta de um utilizador assistente ocorreu o embate e respetiva troca forçada do servo. Com isto não foi possível continuar a afinação do sistema de controlo para as condições existentes, procedendo-se à substituição pelo *Futaba s3003* [24] e assim reiniciando a afinação de parâmetros tendo em conta as novas características deste componente. Optou-se por finalizar o protótipo, retirando-lhe as rodas de apoio e efetuar os ensaios finais. Foi possível perceber que a substituição do servomotor introduz problemas adicionais ao sistema, pois o conjunto tempo de resposta e binário aplicado, são inferiores ao inicialmente planeado. Após a perceção de que os parâmetros de controlo necessitavam de ser alterados, procedeu-se ao respetivo ajuste dos mesmos, iniciando os ensaios nos anteriores à substituição do servomotor. O conjunto que proporcionou o melhor resultado é dado por, Kp = 4.0, Ki = 0.0, Kd = 13. No entanto o resultado é menos satisfatório, uma vez que o sistema não consegue manter o equilíbrio durante o tempo desejado. De referir que estes ensaios foram realizados num troço de estrada com menor inclinação lateral, e assistido por um utilizador de forma a não ocorrer novo embate. A troca imposta no componente da direção é o fator principal do mau resultado, sendo notável o fraco desempenho do *Futaba s3003* [24], tanto no tempo de reação ao impulso gerado no microcontrolador, bem como, e fundamentalmente, o pouco binário aplicado, associado ao atrito gerado no ato de mudar de direção. Os rolamentos internos compostos em plástico são igualmente um fator prejudicial, superior ao já esperado, pois em movimentos bruscos, provocados pelo sinal de controlo, estes acabam por "saltar" não executando o correto movimento angular da direção. Apesar destes aspetos foi possível colocar o sistema em equilíbrio, durante alguns segundos, *Figura 6.9*, variáveis de ensaio para ensaio, mas não conseguindo obter uma afinação ideal do ponto de vista teórico, previsível.

Figura 6.9 - Ensaio Final em Estrada;

Capítulo 7

Implementações Futuras

Este capítulo apresenta recomendações e sugestões de continuação do projeto, onde são incluídas soluções possíveis para resolver alguns dos problemas encontrados. Estas soluções devem ser entendidas como ideias do autor desta tese e não como modelos finais de implementação para o sistema.

7.1 Sistema de Travagem

Um módulo de grande utilidade a ser introduzido ao sistema de forma a aperfeiçoar as viagens autónomas da bicicleta criada, seria o desenvolvimento de um subsistema capaz de efetuar uma travagem rápida e eficaz da mesma. A forma atual de paragem é demasiado rudimentar, onde se espera a sua desaceleração pelo atrito criado. É então possível introduzir duas soluções mais complexas para a resolução deste problema. Usar o conceito de travagem regenerativa, usando o motor, ou simplesmente modificar os simples travões encontrados em todas as bicicletas normais.

7.1.1 Travagem com o motor - Regenerativa

Uma solução interessante e segundo a opinião do autor desta tese, a mais adequada à resolução desta questão, seria a da utilização do motor de tração para a travagem da bicicleta. Esta implementação impõe algumas restrições ao sistema já implementado, sendo uma delas de fácil resolução, ou seja, da necessidade de cancelar o efeito de roda livre da bicicleta, podendo ser resolvida pela alteração do chamado "cubo" do eixo da roda traseira. O problema mais complexo a ser resolvido relacionar-se-ia com o efeito gerado por esta paragem com o motor, denominado por travagem regenerativa. Este sistema é normalmente utilizado em veículos elétricos, onde a energia que é produzida na desaceleração ou travagem do veículo é aproveitada para recarregar as respetivas baterias. Obviamente o motor elétrico consome energia quando necessita de propulsionar o veículo, mas também é capaz de se tornar um gerador de energia quando ajuda na travagem do mesmo. Esta energia necessita de ser transferida para a bateria. Caso isso não seja efetuado, pode ocorrer um sobrecarregamento de energia nos componentes dos circuitos. O atual controlador do motor DC não possui estas características, portanto será necessária a sua substituição por um com a capacidade de lidar com esse efeito. Apesar de ser um circuito relativamente simples de ser construído de raiz, trata-se de algo já comercializado a custos acessíveis, portanto a substituição por um circuito comercial seria a solução ideal.

7.1.2 Servo-Travões

Uma solução simples de travagem que inclui uma mínima alteração ao sistema seria a inclusão de um servomotor, controlado pelo microcontrolador já utilizado, a atuar no cabo dos travões convencionais utilizados nas bicicletas. Este servo, possuindo binário suficiente, é passível de efetuar o movimento de puxar o cabo metálico de um travão, causando assim uma rápida desaceleração do sistema. A relação efeito-custo é aliciante, tornando também o funcionamento do sistema mais idêntico à de uma bicicleta regular.

7.2 Sistema Dinâmico de Parâmetros

A bicicleta estando agora equipada com um controlador funcional para uma determinada velocidade, é capaz de realizar o referido passeio autónomo a essa mesma velocidade constante. Algo que se trata de um modelo um pouco deficiente, pois nem sempre essa é adequada ao cenário de teste pretendido. Será então útil gerar uma espécie de matriz de parâmetros, tornando-os dinâmicos e adaptáveis a uma larga gama de velocidades definidas. Esta matriz pode ser gerada num ambiente de simulações, *Simulink*, e por um algoritmo de *brute force* em *MATLAB* [1], executando o código de forma a obter a matriz de parâmetros do controlador, que posteriormente necessitaria de ser programada no microcontrolador. Obviamente existe um limite inferior e superior para a velocidade, em viagem controlada, sem recorrer à instalação de hardware adicional, contudo seria um melhoramento significativo ao sistema.

7.3 Piloto Automático - Mapeamento

A implementação de um piloto automático, seria uma questão interessante, já abordada inclusive nas simulações, onde se pretende que a bicicleta efetue, durante a viagem de teste, um percurso previamente mapeado. Esta pode ser abordada de diferentes maneiras, utilizando o hardware atual, ou incluindo módulos mais complexos de mapeamento.

7.3.1 Com o sistema atual

Um piloto automático básico, capaz de controlar o movimento da bicicleta na sua navegação inercial poderia ser implementado sem adicionar qualquer hardware ao sistema atual da bicicleta, passando a solução por utilizar um algoritmo de controlo específico. Numa das fases desta tese, a inclusão de algo semelhante foi ponderado, e para tal efetuou-se a simulação em *Simulink* desse módulo. Os resultados apontam para o possível funcionamento em tempo real, sendo o sistema capaz de responder a perturbações externas, seguindo um mapeamento definido previamente. No entanto, este sistema apresenta alguns aspetos negativos, nomeadamente na questão do dito

mapeamento prévio, sendo bastante complicado a introdução de um percurso com pontos mais precisos. A obtenção da posição atual da bicicleta torna-se pouco precisa pois é efetuada através de um mapeamento num sistema de coordenadas, onde se tem conhecimento da taxa de variação e ângulos de direção da bicicleta e a sua velocidade. O percurso seria realizado independentemente do estado físico do cenário de teste no mundo real.

7.3.2 Com GPS

Se um recetor GPS for adicionado ao sistema, seria possível criar uma bicicleta capaz de seguir um caminho definido tendo em conta uma dada posição inicial e final no mundo real. Devido à precisão dos sinais GPS, esta solução necessitaria de uma fusão dos sensores presentes no sistema atual com este sinal GPS.

7.4 Sistema para Evitar Colisões

Um módulo interessante a ser adicionado à bicicleta atual seria o da implementação de um sistema capaz de evitar colisões. Para uma bicicleta poder ser totalmente designada por autónoma, é quase elementar a necessidade de evitar os obstáculos encontrados no seu caminho. As soluções para resolver esta questão poderiam passar pela introdução de câmaras e respetivo software de reconhecimento de imagem, para além de sensores infra vermelhos ou outro tipo de sensores que possuam a capacidade de receber informações do meio que rodeia o sistema. Tratar-se-á de algo demasiado complexo para ser tomado levemente, pois requer um estudo intensivo de outras matérias aqui não analisadas.

7.5 Sistema de Aterragem

O trem de aterragem refere-se a um módulo que introduza ao sistema a capacidade de recuperar, não caindo, quando a velocidade instantânea atingir um limite inferior de controlabilidade, impossibilitando assim o correto funcionamento do controlador de equilíbrio. Este pode ser então usado para levar a bicicleta à velocidade pretendida e para o respetivo processo de paragem da mesma. Para tal é necessário a introdução de hardware adicional ao sistema. Algo capaz de produzir o efeito desejado seria semelhante a um pistão conectado a um conjunto de rodas de apoio, sendo ativado por um porto de saída do microcontrolador, possibilitando assim o equilíbrio ao sistema.

7.6 Alterações de Melhoramento

Dos diversos testes realizados ao sistema foi possível retirar um conjunto de conclusões substanciais sobre o funcionamento de cada componente/módulo incluído à bicicleta inicial. Após respetiva análise, são aqui deixadas algumas considerações sobre pequenas modificações que introduzam melhorias no seu comportamento geral, sendo que algumas destas passam pela simples dedução do que foi visualizado nos ensaios e modelo final, *Figura 7.1*. Antes destas, o ponto essencial passa pela reinstalação do servomotor previamente escolhido, *hexTronic HX12K* (obviamente de um novo), pelos bons resultados obtidos aquando da sua utilização e das repetidas falhas visualizadas do servo substituto.

Figura 7.1 - Ponto de partida nos ajustes do Sistema;

7.6.1 Redução do Peso

Uma característica relevante na atual composição do sistema trata-se do seu peso excessivo comparativamente ao tamanho do mesmo. Por motivos de conveniência de transporte e experimentação, foi escolhida uma bicicleta de criança para modelo inicial do protótipo a compor. O comportamento desta estrutura básica seria idêntico ao de uma bicicleta normal, mas com a introdução dos diversos componentes nesta, provoca um efeito negativo no fator de controlabilidade. Devido ao espaço reduzido para a inclusão de peças com peso significativo, tais como, motor DC e baterias, a localização do centro de massa do sistema final provoca comportamentos mais instáveis na movimentação do mesmo. O peso excessivo destes componentes comparativamente com o peso base da bicicleta provoca de igual modo problemas na aceleração e velocidade máxima da mesma, necessitando de uma potência mais elevada aplicada ao motor. A solução para estes problemas passaria pela relocalização de alguns dos componentes de forma a

centralizar e baixar o centro de massa, e encontrar opções a nível de baterias mais leves. Uma opção mais drástica seria adaptar o sistema a uma bicicleta de tamanho regular, onde a relação de pesos dos componentes adicionais não seria tão substancial.

7.6.2 Aumento da Potência do Motor - Bateria Lítio

Inicialmente o sistema foi pensado e simulado para poder circular a uma velocidade controlada de 4 m/s. Com esta velocidade a gama de parâmetros do controlador seria superior, aumentando a estabilidade da bicicleta. No entanto, foi necessário refazer as simulações para uma velocidade mais reduzida, pois em ensaios preliminares, apenas com o módulo dedicado ativado, concluiu-se que a velocidade máxima atingida é apenas 3.8 m/s. Este aspeto é devido não só ao peso elevado do sistema, como também à falta de potência aplicada ao motor. O último ponto pode ser contraposto utilizando uma bateria capaz de fornecer a potência máxima ao motor melhorando igualmente o tempo de aceleração da bicicleta. As baterias agora usadas seriam então substituídas por uma só, de lítio, capaz de fornecer 21.6 V (6 células) e 10 Ah permitindo assim o motor trabalhar perto da sua máxima potência. Para além destas características as baterias de lítio colmatam o aspeto do peso extra, uma vez que o seu peso é inferior às regulares de ácido-chumbo, pesando apenas cerca de 2 kg.

7.6.3 Peça de Direção

A peça de direção que efetua a conexão entre o veio da forqueta e o servomotor foi desenhada tendo em conta noções de sistemas compostos em veículos tipo karts. Esta peça possibilita a movimentação através de um único ponto de contacto com o servo motor, limitando-lhe de igual modo a excursão de graus aplicados. Apesar da boa resposta aplicada pelo módulo, este poderia ser ligeiramente melhorado, utilizando para isso uma peça composta pela atual e uma outra simétrica. Assim existiriam dois pontos de ligação entre o servomotor e o veio de direção, aumentando a robustez do módulo e assim ficando mais semelhante com as atuais direções dos karts, como pode ser visto na *Figura 7.2*.

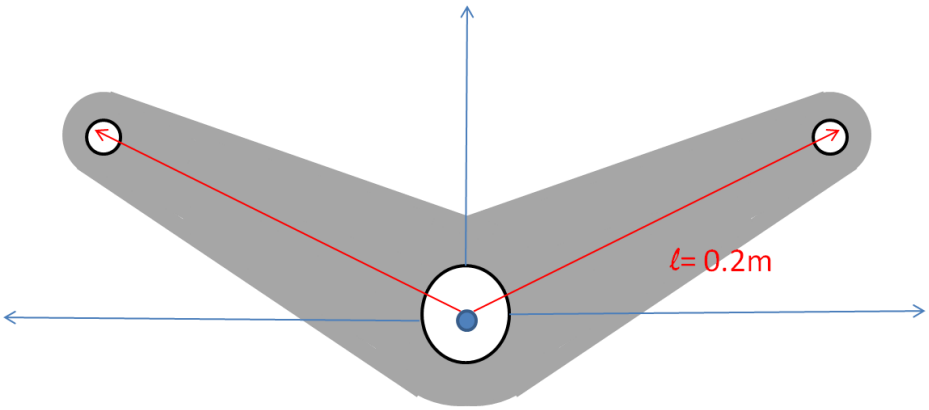


Figura 7.2 - Peça Direção Modificada;

De forma a aumentar a excursão dos graus provocados na direção, pelo servo, o comprimento desta peça, deveria passar dos atuais 0.12 m para 0.2 m, podendo conjugar com o comprimento do conetor entre os dois conjuntos.

7.6.4 Controlador LQR com Estimador de Kalman

Após a conclusão dos ensaios utilizando o controlador PID, foi possível reparar que o seu comportamento é passível de ser melhorado. Assim, a utilização de um controlador com melhor "rendimento" é um aspeto a ser ponderado. Um controlador possível trata-se do LQR, sendo este uma possível solução a implementar no sistema. Conjugado com um estimador de *Kalman,* de forma a estimar estados não medidos, conseguirá controlar o equilíbrio da bicicleta de um modo mais eficaz, visto esta apresentar características essenciais a sistemas com comportamentos extremamente dinâmicos. É previsível que a implementação deste controlador LQR seja possível e assim ser capaz de apresentar uma melhor resposta comparativamente ao PID, prevendo o comportamento e inércia do sistema.

7.6.5 Controlo de Velocidade e Tacómetro

Um dos módulos implementados na bicicleta foi o do controlador de velocidade e respetivo tacómetro, para a medição da velocidade instantânea da mesma. Em ensaios preliminares detetou-se que a velocidade máxima atingida pela bicicleta, como já mencionado, não excedia os 3.8 m/s. Este fator reduziu a exigência em ser elaborado um controlador ideal, direcionando assim o foco apenas no correto controlo de equilíbrio. Apesar de funcional, assim que ocorreu um aumento da potência máxima do motor, os parâmetros do controlador PI, responsável pelo controlo de velocidade, necessitaram de ajustes para cumprir com as exigências requeridas. Obviamente este controlador não precisa de levar e manter a bicicleta a uma velocidade exata, podendo assumir uma gama de valores para os quais os mesmos parâmetros de equilíbrio conseguem manter a bicicleta em funcionamento. Em relação ao tacómetro a alteração foi efetuada durante a execução do protótipo, onde a primeira solução pensada foi a de um sensor de *Hall*, substituído por um com a tecnologia infravermelho, capaz assim de aumentar a precisão do módulo. Fica à consideração a colocação deste, noutro local que se considere mais adequado.

7.7 Sistema de Massas Variáveis - Pêndulo Invertido

Um módulo adicional com a capacidade de aumentar a gama de equilíbrio do protótipo criado, consistiria num sistema de massas variáveis. Seria então montado no quadro da bicicleta uma estrutura que conseguisse variar o centro de massa daquela, de forma a mantê-la equilibrada, possivelmente até mesmo, quando o sistema se encontre parado. Utilizando um código extra, semelhante ao já existente para o controlo recorrendo ao eixo de direção, seria medida a inclinação da bicicleta atuando-se neste novo módulo. Esta adição é útil para a construção de um sistema de mapeamento, sem a necessidade de ser a direção a manter o equilíbrio da bicicleta. Esta ideia é então baseada na implementação de um pêndulo invertido. Um conceito já existente e descrito sumariamente no *ponto 1.3.1.* A *Figura 7.3* pretende representar a esquematização desta implementação no modelo atual da bicicleta.

Figura 7.3 - Esquema Bicicleta com Pêndulo Invertido;

Capítulo 8

Conclusão e Discussão de Resultados

Após a apresentação dos resultados obtidos, serão aqui discutidos alguns aspetos que poderiam ter sido realizados de maneira diferente no decorrer da realização desta tese e quiçá, tenham tido alguma influência nos mesmos. Será de igual modo feita uma breve comparação das simulações realizadas em ambiente *Simulink* com o que foi visto em funcionamento real.

8.1 Medição dos Parâmetros da Bicicleta

Em relação às medições realizadas à bicicleta, de modo a obter os seus parâmetros para o modelo simulado, não teria sido possível efetuá-las com uma maior precisão, dadas as ferramentas existentes e a imprecisão característica dos diversos componentes de uma bicicleta. Considera-se que os erros de medição são insignificantes em comparação com alguns erros causados pelas suposições feitas nas equações linearizadas do modelo da bicicleta. Foi então conseguido obter os parâmetros representativos da bicicleta, construindo assim um modelo capaz de simular o sistema em computador.

8.2 Simulações

Após a construção de um bloco que modela o sistema final, recriou-se ambientes de simulação capazes de representar situações do mundo real, que ajudem à compreensão do funcionamento do mesmo e validem um conjunto de parâmetros do controlador que servem como ponto de partida para a implementação do controlador no sistema real. Um controlador PID foi o escolhido para a realização do projeto. Este controlador, apesar de simples, após a afinação mostrou que é possível equilibrar a bicicleta para uma velocidade fixa. A velocidade teve de ser adaptada a valores capazes de serem reproduzidos no modelo real. O *Setpoint* de equilíbrio é perfeitamente controlado, apesar de ocorrerem pequenas oscilações, podendo estas ser consideradas normais dado que se incluíram na simulação perturbações que sempre existirão numa implementação real. Para além da análise do equilíbrio é também possível analisar a excursão do deslocamento lateral (considerando uma linha reta desde o ponto de partida), bem como os ângulos que a direção terá de sofrer de forma a manter a inclinação pretendida. Foram testados vários cenários tentando cobrir as dificuldades que o sistema possa encontrar, incluindo também simulações para futuros projetos. Os bons resultados nas simulações deram luz verde à realização de testes reais.

8.3 Equilíbrio da Bicicleta no Mundo Real

Analisando os cenários de testes produzidos em ambiente de simulação tentou-se reproduzi-los no mundo real. Para tal, recriou-se primeiramente um ensaio ideal, num pavilhão gimnodesportivo, onde não são aplicadas forças externas de elevada magnitude. Assim, foi possível concluir acerca da funcionalidade do sistema, num teste com um grau de menor dificuldade. Dessa maneira, foram encontrados os parâmetros do controlador, para que o sistema se comportasse como delineado. Os módulos implementados na bicicleta cumpriram a função traçada, concluindo assim o objetivo principal desta tese, onde se efetuou uma viagem autónoma em equilíbrio com comportamento semelhante ao visualizado nas simulações. O módulo de controlo de velocidade acabou em muitos ensaios por funcionar em malha aberta devido à falta de potência aplicada ao motor. Apesar de funcional, tornou-se inclusive preferível aplicar a tensão máxima disponível ao motor DC, já que a velocidade terminal se encontrava na gama dos parâmetros de controlo de equilíbrio encontrados. Foi também possível assistir a uma pequena assimetria na constituição da bicicleta, alterando ligeiramente o comportamento do centro de massa. Esta falta de simetria é devida à instalação necessariamente assimétrica de alguns componentes no corpo do sistema. Após cumprido o objetivo fundamental num cenário de testes ideal, colocou-se o sistema em funcionamento numa estrada. Com os parâmetros de controlo obtidos dos ensaios realizados em pavilhão efetuaram-se novos testes resultando na quebra irreparável do servomotor previamente designado, isto após alguns ensaios com resultados positivos. Este acidente poderia ter sido previsto, como foi demonstrado numa simulação, tendo esta já em conta a inclinação da estrada. O facto de apenas se controlar o equilíbrio e não a direção, com a inclinação natural da estrada, o sistema acaba por a compensar virando a direção rumo à berma. Após a troca do servomotor original por um com menores binário e tempo de reação os resultados obtidos foram mais modestos, mas mesmo assim verificou-se ser possível controlar a bicicleta durante vários segundos. Comprovou-se pois a viabilidade do protótipo e dos conceitos teóricos, bem como a concretização da ideia do projeto inicial. No entanto este sistema apresenta debilidades que podem ser futuramente corrigidas, como já foi descrito no *Capitulo 7*.

8.4 Conclusão

O trabalho realizado durante esta tese foi realizado com o objetivo principal de modificar uma bicicleta regular de criança, transformando-a numa bicicleta autónoma, capaz de realizar um certo deslocamento, a uma velocidade pré-determinada, num sistema de autoequilíbrio. O sistema projetado e implementado, quer de hardware, como de software, para a bicicleta é então capaz de cumprir, com um desempenho adequado, esse objetivo. Embora apresente debilidades, o sistema consegue cumprir com as especificações exigidas sendo capaz de executar uma viagem equilibrada em diferentes cenários de teste.

Bibliografia

[1] MathWorks, “MatLab.” MathWorks, 2012.
[2] H. O. Loftum, “Styresystem for kybernetisk sykkel - instrumentering for styring av en tohjuls herresykkel,” no. June, 2006.
[3] L. Bjermeland, “Modeling, simulation and control system for an autonomous bicycle,” no. June, 2006.
[4] J. A. Fossum, “Instrumentering og datasystemer for autonom kybernetisk sykkel,” no. December, Trondheim, 2006.
[5] A. Sølvberg, “CyberBike,” no. June, 2007.
[6] S. E. Brekke, “Autonomous Bicycle,” *Sci. Technol.*, no. September, 2010.
[7] D. B. Hatlevoll, “Autonomous bicycle, hardware and software development,” no. January, 2011.
[8] D. C. Ånnestad, “Autonomous Bicycle,” *Sci. Technol.*, no. September, 2010.
[9] James Neill of HNH Partners Ltd, “JYROBIKE LTD,” 2014.
[10] J. J. W. Whipple, “The stability of the motion of a bicycle,” *J. Pure Appl. Math.*, 1899.
[11] J. M. Papadopoulos, “Bicycle steering dynamics and self-stability: a summary report on work in progress,” Ithaca, NY, 1987.
[12] Peugeot, “Peugeut Bicycle,” 2016. [Online]. Available: http://img.turbo.fr/02599386-photo-peugeot-velo-a-assistance-electrique-vae.jpg. [Accessed: 20-Aug-2016].
[13] G. Interface, “Pic32mx5xx/6xx/7xx,” 2013.
[14] Microchip, “MPLAB X.” Microchip, 2014.
[15] C. Knospe, “PID control,” *Control Syst. IEEE*, vol. 26, no. 1, pp. 216–251, 2006.
[16] CadSoft, “Eagle.” Cadsoft, 2016.
[17] HexTronic, “HX12K Servo.” HexTronic, 2015.
[18] I. Biometrics, “xx555 Precision Timers PACKAGE.” 2014.
[19] B. Ave, D. Number, and R. Date, “MPU-6000 and MPU-6050 Product Specification,” 2013.
[20] Texas Instruments, “LMx93-N , LM2903-N Low-Power , Low-Offset Voltage , Dual Comparators,” pp. 1–20, 2014.
[21] MyScratchBooks, “IR sensor,” 2016. [Online]. Available: https://sites.google.com/site/myscratchbooks/_/rsrc/1420953044019/home/projects/project-11-infrared-speed-sensing-module/lm393.jpg. [Accessed: 25-Sep-2016].
[22] G. Description, “LM317L 3-Terminal Adjustable Regulator LM317L 3-Terminal Adjustable Regulator,” no. March, 2002.
[23] N. H. Getz and J. E. Marsden, “{C}ontrol for an autonomous bicycle, {P}aper 525473,” *Int. Conf. Robot. Autom.*, vol. 2, pp. 1397–1402, 1995.
[24] S. F. Servo, “S3003 futaba servo,” p. 3003, 2014.

Apêndice A

Código e Testes Aplicados

A.1 Funções de Teste

```
// Parameters for PID controller**********************************************/

static float Kp = 4.1;
static float Ki = 0.002;
static float Kd = 12;

static float offs = -2.3;       //Offset to maintain balance*********************************/

static float Steer = 0.0;
static float v =  0.0;

// Parameters for PI DC motor controller*************************************/

static float KpDC = 10;
static float KiDC = 0.05;
static float vel = 3.8;        //Desire velocity*************************************/

static float i_anteriorDC;
static float erro_anteriorDC;
```

```
void Equilibrium(void)
{
  /*Funtion Equilibrium()
   *********************
   * Run the funtion to maintain the equilibrium of the bike;
   *********************
   */

  //********Initialization's For EQUILIBRIUM Control*********************************/
  //********Setup's*****************************************************/
  Setup_I2C();
  Setup_MPU6050();
  Setup_DCmotor();
  Setup_PWM();
```

```
  Setup_Timer4();
  Zero_Sensors();
  INTEnableSystemMultiVectoredInt();

  //*********Initial Condition's*************************************************/
  float act = 0;
  i_anterior = 0;
  erro_anterior = 0;
  Calibrate_Gyros();
  int op = 0;

  do {
    printf("\nOP1 - EQUILIBRIUM; OP2 - CHANGE PID PARA; OP3 - CHANGE OFFS; 4 - EXIT; ");
    op = readIntN();

    if (op == 1) {
      i_anterior = 0;
      erro_anterior = 0;
      c = 0;

      while (U1STAbits.URXDA != 1) {
        if (flag == 1) {
          Get_Accel_Values();
          Get_Accel_Angles();
          act = ACCEL_YANGLE;
          PID(act, offs);              //ANGLE offs, offset in equilibrium*******************/
          }
          flag = 0;
        }
    }
     else if (op == 2) {
      ChangePIDparameters();
    }
     else if (op == 3) {
      printf("\nOffset atual = %f, new Offset? ", offs);
      offs = readFloat();
    }
     else if (op == 4) {
      T4CONbits.ON = 0;
    }
  } while (op != 4);
}
```

```
void ServoCopy (void)
{
  /*Funtion ServoCopy()
   ********************
    * Servo Motor copy the angle measured in the MPU;
   ********************
   */

  //*******Setup's************************************************************/
  Setup_I2C();
  Setup_MPU6050();
  Setup_PWM();
  Setup_Timer4();
  Zero_Sensors();
  INTEnableSystemMultiVectoredInt();

  //*******Initial Condition's*************************************************/
  float act = 0;
  int op;
  unsigned int c = 0;

  do{
  printf("\nOPTION 1 - Servo Copy; OPTION 2 - EXIT; ");
  op = readIntN();

  if(op == 1)
  {
  c=0;
  while(c<10000000)
  {
        if(flag != 0)
        {
    Get_Accel_Values();
    Get_Accel_Angles();
    act = ACCEL_YANGLE;
    printf("\n %f \n ",act);

       if (flag == 1)
         {
           viraMotor(act);
         }
    }
    c++;
```

```
    flag=0;
  }
  }
  else
  {
    T4CONbits.ON = 0;
  }
  }while(op!=2);
}
```

```
void TesteLEITURA(void)
{
  /*Funtion TesteLEITURA()
   ********************
    * Tests the reading of the MPU;
   ********************
   */

    //***Setup's*************************************************************/
    Setup_I2C();
    Setup_MPU6050();
    Zero_Sensors();

    //***Initial Condition***************************************************/
    unsigned char Data = 0x00;
    unsigned char c;

    //***I2C Verification****************************************************/
    LDByteReadI2C(MPU6050_ADDRESS, MPU6050_RA_WHO_AM_I, &Data, 1);
    printf("\nMPU6050 Address: 0x%x \n", Data);
    MPU6050_Test_I2C();

    //***Calibration's*******************************************************/
    //Calibrate_Gyros();


    while(c!=32)  //SPACE TO STOP*****************************************/
    {
      c = readChar();
      //printf("\nGyro Values: \n");
      //Get_Gyro_Rates();

      printf("\nAccelarometer Values:\n");
```

```
        Get_Accel_Values();
        Get_Accel_Angles();
        printf("\n\nAcel->Angulo EixoX: %f \nAcel->Angulo EixoY: %f", ACCEL_XANGLE,
         ACCEL_YANGLE);
    }
}
```

```
void TesteLEITURA_gyro_accel_combin(void)
{
  /*Funtion TesteLEITURA_gyro_accel_combin()
   ********************
    * Tests the reading of the MPU with gyro+accel combined;
   ********************
   */
     //***Setup's*****************************************************************/
     Setup_I2C();
     Setup_MPU6050();
     Zero_Sensors();
     //Setup_Timer1();


     //***Initial Condition*******************************************************/
     unsigned char Data = 0x00;
     unsigned char c;

     //***I2C Verification********************************************************/
     LDByteReadI2C(MPU6050_ADDRESS, MPU6050_RA_WHO_AM_I, &Data, 1);
     printf("\nMPU6050 Address: 0x%x \n", Data);
     MPU6050_Test_I2C();

     //***Calibration's***********************************************************/
     Calibrate_Gyros();


     while(c!=32)  //SPACE TO STOP*********************************************/
     {
        c = readChar();
        COMPLEMENT_Sensors();
     }
}
```

```
void TesteSERVO(void)
{
   /*Funtion TesteSERVO()
   ********************
    * Tests the ServoMotor;
   ********************
   */
     //***Setup's*************************************************************/
     Setup_PWM();
     //INTEnableSystemMultiVectoredInt();
     //***Initial Condition ***************************************************/
     int graus =  0;
     int c = 0;

     do
     {
     printf("\nServo Test: 1 - Fast Test; 2  - Choose Angle; 3 - EXIT \n");
     c=readIntN();
     if(c==1)
     {
     //***Fast Test***********************************************************/
     DelayMs(1000);
     viraMotor(45.0);
     printf("\n45 ");
     DelayMs(1000);
     viraMotor(0.0);
     }
     else if(c == 2)
     {
     //*** Choose angle*******************************************************/
     printf("\nNote: Type 99 to leave!");
     printf("\nNumber of degrees desired:");
        graus = readIntN();
     while (graus != 99)
        {
         viraMotor((float)graus);
         printf("\nNumber of degrees desired: ");
         graus = readIntN();
        }
     }
     }while(c!=3);
}
```

```
float getV()
{
  return Veloc;
}
```

```
void TesteDCmotor(void)
{
   /*Funtion TesteSERVO()
   ********************
    * Tests the DC Motor;
   ********************
   */
     //***Setup's*****************************************************************/
     Setup_DCmotor();
     float v = 0.0;
     //*** Choose angle***********************************************************/
     printf("\nNote: Type 99 to leave!");
     printf("\nPWM desired: ");
        v = readFloat();
     while (v != 99)
        {
         velocControl((float)v);
         printf("\nPWM: ");
         v = readFloat();
        }
}
```

```
void TesteDCmotorControl(void)
{
   /*Funtion TesteDCmotor()
   ********************
    * Tests the function to control DC motor velocity;
   ********************
   */
     //***Setup's*****************************************************************/
     Setup_DCmotor();
     Setup_Timer4();
     setup_Tach();
     initLANDING();
     INTEnableSystemMultiVectoredInt();
```

```
    //***Initial Condition*****************************************************/
    float v =  0.0;

    while(1)
    {
      if(flag==1)
      {
       v = getV();
       printf("Vel: %f",v );

       if(v>vel-1.0)    //If velocity > 3.0 m/s disable landing kit*************************/
          TurnOFF();
       else
          TurnON();

       PI(v,vel);
      }
      flag=0;
    }
}
```

```
void TestTach()
{
  /*Funtion TestTach()
   ********************
    * Test the tachometer;
   ********************
   */
  setup_Tach();

  while(1)
  {
    printf("\nVelocity = %f",getV());
    DelayMs(10000);
  }
}
```

A.2 Função Main - Final

```
int main (void)
{
    // FINAL RUN CODE

   SYSTEMConfigPerformance(SYSCLK);
   mOSCSetPBDIV(OSC_PB_DIV_2);
   INTDisableInterrupts();

   // Initialization's
   initUART();
   Setup_I2C();
   Setup_MPU6050();
   Setup_PWM();
   Setup_Timer4();
   Zero_Sensors();
   Setup_DCmotor();
   setup_Tach();
   INTEnableSystemMultiVectoredInt();

   // Parameters
   float act = 0;
   i_anterior = 0;
   erro_anterior = 0;
   float actVel = 0;
   int time = 200000;
   TurnON();

   while (time != 0)
   {
      if (flag == 1)
      {
         actVel = getV();       //Read actual velocity*************************************/
         PI(actVel,vel);

         if(actVel > 3.0)       //Control equilibrium only at this level*********************/
         {
         Get_Accel_Values();
         Get_Accel_Angles();
         act = ACCEL_YANGLE;
         PID(act, offs);        //Angle Offset to equilibrium*********************************/
```

```
        TurnOFF();          //Turn Off the Landing Kit************************************/
        }
        time--;             //Decrement the time counter*********************************/
        TurnON();
      }
      flag = 0;
   }

   TurnON();                //Turn on the Landing Kit************************************/
   velocControl(0);         //Turn Off the DC motor*************************************/

return 0;
}
```

Apêndice B

Rotinas e Simulações

B.1 Rotina para obter Centro Massa [8]

```
%GENERAL PARAMETERS

w = 0.755;              % Wheelbase
r_R = 0.1939;           % Radius rear wheel
r_F = 0.1935;           % Radius front wheel
alpha = 59*pi/180;      % head tube angle
lambda = 31*pi/180;     % Steer axis tilt
lambda_st = 72*pi/180;  % seat tube angle
lambda_tt = 13*pi/180;  % top tube angle
lambda_bt = 46*pi/180;  % bottom tube angle

%REAR WHEEL
cm_R = [0 0 -r_R];
%FRONT WHEEL
cm_F = [w 0 -r_F];
%REAR BODY FRAME
a_B = [-0.101;0.256; 0.299];

b_B = [197*pi/180;103*pi/180;328*pi/180];

b = -[a_B(1)/cos(b_B(1))+r_R; a_B(2)/cos(b_B(2))+r_R;
a_B(3)/cos(b_B(3))+r_R];

m = -[tan(b_B(1));tan(b_B(2));tan(b_B(3))];

xz_a = inv([-m(1) 1;-m(2) 1])*[b(1);b(2)];
xz_b = inv([-m(2) 1;-m(3) 1])*[b(2);b(3)];
xz_c = inv([-m(1) 1;-m(3) 1])*[b(1);b(3)];

cm_B = [(xz_a(1)+xz_b(1)+xz_c(1))/3 0 (xz_a(2)+xz_b(2)+xz_c(2))/3];


%FRONT HANDLEBAR FRAME
a_H = [0.324;-0.006];

b_H = [(22*pi/180);(295*pi/180)];

b = [w*tan(b_H(1))-r_F-a_H(1)/cos(beta(1,3)); w*tan(b_H(2))-r_F-
a_H(2)/cos(beta(2,3))];

m = [-tan(b_H(1));-tan(b_H(2));];

xz = inv([-m(1) 1;-m(2) 1])*[b(1);b(2)];
```

```
cm_H = [xz(1) 0 xz(2)];
```

B.2 Rotina para obter Momento Inércia [8]

```
%GENERAL PARAMETERS

g = 9.81;               % Gravity
w = 0.755;              % Wheelbase
m_R = 1.340;            % Mass rear wheel
m_H = 1.606;            % Mass front frame
m_F = 1.134;            % Mass front wheel
lambda_st = 72*pi/180;  % seat tube angle
lambda_tt = 13*pi/180;  % top tube angle
lambda_bt = 46*pi/180;  % bottom tube angle

%Calibration
m_calib = 0.365;
l_calib = 1.000;
r_calib = 0.0050;
I_calib = m_calib/12*(3*r_calib^2+l_calib^2);
T_calib = (14.95+15.61+14.30)/10/3;
k = (4*I_calib*pi^2)/(T_calib^2);

%Rear WHEEL
nu_R = 20;
l_R = 0.136;
t_R_xz = (19.46+19.91+19.52)/3/nu_R;
t_R_y = (19.70+19.65+19.53)/3/nu_R;
I_R_xx = k*t_R_xz^2 / (4*pi^2);
I_R_yy = ((t_R_y/(2*pi))^2*(m_R*g*l_R))-(m_R*l_R^2);
I_R_zz = I_R_xx;
I_R = [I_R_xx I_R_yy I_R_zz];

%Rear Frame
nu_B = 20;                          %Number of oscillation
t_B =
[(41.75+42.31+42.10)/3/nu_B;(58.73+58.61+58.82)/3/nu_B;(58.65+56.32+57.25
)/3/nu_B];
J_B = [k*t_B(1)^2/(4*pi^2);k*t_B(2)^2/(4*pi^2);k*t_B(3)^2/(4*pi^2)];
b_B = [197*pi/180;103*pi/180;328*pi/180];


%Transformation matrix rear frame
JI_B = [cos(b_B(1))^2 -2*sin(b_B(1))*cos(b_B(1)) sin(b_B(1))^2;
cos(b_B(2))^2 -2*sin(b_B(2))*cos(b_B(2)) sin(b_B(2))^2; cos(b_B(3))^2 -
2*sin(b_B(3))*cos(b_B(3)) sin(b_B(3))^2];
I_Bt = JI_B\J_B;
I_B = [I_Bt(1) 0 I_Bt(2); 0 0 0;I_Bt(2) 0 I_Bt(3)];

%Front Frame
cm_H = [0.9015 0 -0.7223];
r_F = 0.1935;
l_H = sqrt((cm_H(1)-w)^2+(cm_H(3)+r_F)^2);
nu_H = [10;10;10;20];
```

```
t_H_xz =
[(25.13+25.31+25.46)/3/nu_H(1);(17.09+18.37+18.20)/3/nu_H(2);(20.12+20.40
+20.23)/3/nu_H(3)];
t_H_y = (20.99+20.61+20.72)/3/nu_H(4);
J_H =
[k*t_H_xz(1)^2/(4*pi^2);k*t_H_xz(2)^2/(4*pi^2);k*t_H_xz(3)^2/(4*pi^2);];

b_H = [22*pi/180;295*pi/180;153*pi/180];

%Front frame
JI_H = [cos(b_H(1))^2 -2*sin(b_H(1))*cos(b_H(1))
sin(b_H(1))^2;cos(b_H(2))^2 -2*sin(b_H(2))*cos(b_H(2)) sin(b_H(2))^2;
cos(b_H(3))^2 -2*sin(b_H(3))*cos(b_H(3)) sin(b_H(3))^2];
I_H_xz = JI_H\J_H;
I_H_y = ((t_H_y/(2*pi))^2*(m_H*g*l_H))-(m_H*l_H^2);
I_H = [I_H_xz(1) 0 I_H_xz(2); 0 I_H_y 0;I_H_xz(2) 0 I_H_xz(3)];
```

B.3 Rotina com os Parâmetros do Modelo

```
%Front wheel
nu_F = 20;                                          %Number of osc
l_F = 0.136;                                        %m length cpend
t_F_xz = (18.87+19.02+19.13)/3/nu_F;                %s Avg time, tpend
t_F_y = (20.99+20.61+20.72)/3/nu_F;                 %s Avg time, cpend
I_F_xx = k*t_F_xz^2 / (4*pi^2);                     %kgm^2 MOI, x and z?axis
I_F_yy = ((t_F_y/(2*pi))^2*(m_F*g*l_F))-(m_F*l_F^2);
I_F_zz = I_F_xx; I_F = [I_F_xx I_F_yy I_F_zz];

%General parameters
w = 0.755;                          %Wheelbase(m)
c = 0.115;                          %trail(m)
alpha = 59*pi/180;                  %head angle(rad)
lambda = 31*pi/180;                 %Steer axis til (rad)
g = 9.81;                           %gravity(N/kg)

%Rear wheel
cm_R = [0 0 -0.1939];               %center of mass
r_R = 0.1939;                       %radius(m)
m_R = 1.340;                        %mass(kg)
I_R = [0.0131 0.0188 0.0131];       %moment of inertia

%Rear body frame
cm_B = [-0.1350 0 -0.4653];          %center of mass
m_B = 7.915;                         %mass(kg)
I_B = [0.0753 0 0.0317; 0 0 0;0.0317 0 0.1049]; %moment of inertia

%Front handlebar frame
cm_H = [0.6181 0 -0.4811];          %center of mass
m_H = 1.606;                        %mass(kg)
I_H = [0.0763 0 -0.0199;0 0 0;-0.0199 0 0.0549]; %moment of inertia
```

```
%Front wheel
cm_F = [0.7550 0 -0.1935];      %center of mass
r_F = 0.1935;                   %radius(m)
m_F = 1.134;                    %mass(kg)
I_F = [0.0123 0.0204 0.0123];   %moment of inertia
```

B.4 Rotina para obter o Modelo da Bicicleta [8]

```
clc; clear;
run ./Parameters
disp('bicycleparameters OK')

%Creating variables from bicycle

la = lambda;
x_B = cm_B(1); z_B = cm_B(3); x_H = cm_H(1); z_H = cm_H(3);
I_Rxx = I_R(1,1); I_Ryy = I_R(1,2); I_Rzz = I_R(1,3);
I_Bxx = I_B(1,1); I_Bxz = I_B(1,3); I_Bzz = I_B(3,3);
I_Hxx = I_H(1,1); I_Hxz = I_H(1,3); I_Hzz = I_H(3,3);
I_Fxx = I_F(1,1); I_Fyy = I_F(1,2); I_Fzz = I_F(1,3);

%Calculating parameters with equations

m_T = m_R+m_B+m_H+m_F;
%(eq.2.3)
x_T = (x_B*m_B+x_H*m_H+w*m_F)/m_T;
%(eq.2.4)
z_T = (-r_R*m_R+z_B*m_B+z_H*m_H-r_F*m_F)/m_T;
%(eq.2.5)
I_Txx = I_Rxx+I_Bxx+I_Hxx+I_Fxx+m_R*r_R^2+m_B*z_B^2+m_H*z_H^2+m_F*r_F^2;
%(eq.2.6)
I_Txz = I_Bxz+I_Hxz-m_B*x_B*z_B-m_H*x_H*z_H+m_F*w*r_F;
%(eq.2.7)


%eq.2.8
I_Tzz = I_Rzz+I_Bzz+I_Hzz+I_Fzz+m_B*x_B^2+m_H*x_H^2+m_F*w^2;
%(eq.2.9)

m_A = m_H+m_F;
%(eq.2.10)
x_A = (x_H*m_H+w*m_F)/m_A;
%(eq.2.11)
z_A = (z_H*m_H-r_F*m_F)/m_A;
%(eq.2.11)

I_Axx = I_Hxx+I_Fxx+m_H*((z_H-z_A)^2)+m_F*((r_F+z_A)^2);
%(eq.2.12)
I_Axz = I_Hxz-m_H*(x_H-x_A)*(z_H-z_A)+m_F*(w-x_A)*(r_F+z_A);
%(eq.2.13)
I_Azz = I_Hzz+I_Fzz+m_H*((x_H-x_A)^2)+m_F*((w-x_A)^2);
%(eq.2.14)
```

```
u_A = (x_A-w-c)*cos(la)-z_A*sin(la);
%(eq.2.15)


I_ALL =
m_A*u_A^2+I_Axx*sin(la)^2+2*I_Axz*sin(la)*cos(la)+I_Azz*cos(la)^2;
%(eq.2.16)
I_ALx = -m_A*u_A*z_A+I_Axx*sin(la)+I_Axz*cos(la);
%(eq.2.17)
I_ALz = m_A*u_A*x_A+I_Axz*sin(la)+I_Azz*cos(la);
%(eq.2.18)

mu = c*cos(la)/w;
%(eq.2.19)
S_R = I_Ryy/r_R;
%(eq.2.20)
S_F = I_Fyy/r_F;
%(eq.2.20)
S_T = S_R+S_F;
%(eq.2.20)
S_A = m_A*u_A+mu*m_T*x_T;
%(eq.2.21)

M = zeros(2);
%(eq.2.23)
K0 = zeros(2);
%(eq.2.25)
K2 = zeros(2);
%(eq.2.27)
C1 = zeros(2);
%(eq.2.29)

M(1,1) = I_Txx;
%(eq.2.22a)
M(1,2) = I_ALx+mu*I_Txz;
%(eq.2.22a)
M(2,1) = M(1,2);
%(eq.2.22b)
M(2,2) = I_ALL+2*mu*I_ALz+mu^2*I_Tzz;
%(eq.2.22b)

K0(1,1) = m_T*z_T;
%(eq.2.24a)
K0(1,2) = -S_A;
%(eq.2.24a)
K0(2,1) = K0(1,2);
%(eq.2.24b)
K0(2,2) = -S_A*sin(la);
%(eq.2.24b)

K2(1,1) = 0;
%(eq.2.26a)
K2(1,2) = (S_T-m_T*z_T)*cos(la)/w;
%(eq.2.26a)
K2(2,1) = 0;
%(eq.2.26b)
```

```
K2(2,2) = (S_A+S_F*sin(la))*cos(la)/w;
%(eq.2.26b)

C1(1,1) = 0;
%(eq.2.28a)
C1(1,2) = mu*S_T+S_F*cos(la) +I_Txz*cos(la)/w-mu*m_T*z_T;
%(eq.2.28a)
C1(2,1) = -(mu*S_T+S_F*cos(la));
%(eq.2.28b)
C1(2,2) = I_ALz*cos(la)/w+mu*(S_A+I_Tzz*cos(la)/w);
%(eq.2.28b)
disp('initModel OK');
save whipple g M K0 K2 C1;
disp('Parameters saved to whipple');
```